Feature Synthesized EM Algorithm for Image Retrieval

RUI LI, BIR BHANU, and ANLEI DONG
University of California, Riverside

As a commonly used unsupervised learning algorithm in *Content-Based Image Retrieval* (CBIR), *Expectation-Maximization* (EM) algorithm has several limitations, including the curse of dimensionality and the convergence at a local maximum. In this article, we propose a novel learning approach, namely *Coevolutionary Feature Synthesized Expectation-Maximization* (CFS-EM), to address the above problems. The CFS-EM is a hybrid of *coevolutionary genetic programming* (CGP) and EM algorithm applied on partially labeled data. CFS-EM is especially suitable for image retrieval because the images can be searched in the synthesized low-dimensional feature space, while a kernel-based method has to make classification computation in the original high-dimensional space. Experiments on real image databases show that CFS-EM outperforms *Radial Basis Function Support Vector Machine* (RBF-SVM), CGP, *Discriminant-EM* (D-EM) and *Transductive-SVM* (TSVM) in the sense of classification performance and it is computationally more efficient than RBF-SVM in the query phase.

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Coevolutionary feature synthesis, expectation maximization, semi-supervised learning, content-based image retrieval

ACM Reference Format:

Li, R., Bhanu, B., and Dong, A. 2008. Feature synthesized EM algorithm for image retrieval. ACM Trans. Multimedia Comput. Commun. Appl. 4, 2, Article 10 (May 2008), 24 pages. DOI = 10.1145/1352012.1352014 http://doi.acm. org/10.1145/1352012. 1352014

1. INTRODUCTION

The handling of high feature dimensionality and the labeling of training data are the two major challenges in *content-based image retrieval* (CBIR). The supervised learning, for example, with its two representative approaches, *support vector machine* (SVM) algorithm [Vapnik 2000] and boosting algorithm [Meir and Rtsch 2002], can often overcome the "curse of dimensionality" in many pattern recognition applications. However, it is unrealistic to obtain a large amount of labeled data in CBIR due to the high cost of data collection and the subjectivity of human image perception. On the other hand, unsupervised learning, for example, the *Expectation Maximization* (EM) algorithm which is commonly used in CBIR does not have the labeling problem, but it needs a large amount of data especially when the feature dimension is high, which is not always available. Thus, neither supervised learning nor unsupervised

The research was partially supported by NSF Information Technology Research Grant 0114036. The contents of the information do not reflect the position or policy of the U.S. government.
Authors' address: Center for Research in Intelligent Systems, University of California, Riverside, CA 92521; email: {rli,bhanu,adong}@vislab.ucr.edu.

learning can solve both the high dimensionality and the labeling problems simultaneously. Therefore, the learning for image databases usually can only be carried out with the hybrid of both labeled and unlabeled data, called *transductive learning* [Virginia 1993].

The basic idea behind our transductive learning is that a feature transformation and a classifier can be learned from the labeled data. The unlabeled data are transformed using the learned transformation and then classified using the learned classifier. EM algorithm is used on both the labeled and unlabeled training data to refine the classifier. Since all the data are now "labeled" (not necessarily perfect labels), the feature transformation and the classifier are refined iteratively.

As the learning on the labeled data plays a key role in this process, we need to find an appropriate feature transformation to deal with the high feature dimensionality of image databases. *Fisher discriminant analysis* or *multiple discriminant analysis* (MDA) is a well-known approach to obtain discriminant features by using a linear transformation. Wu et al. [2000] integrate MDA with EM algorithm to develop a *Discriminant-EM* (D-EM) method. It is based on the assumption of Gaussian distribution and it shows good results for image retrieval. Dong et al. [2005] describe a *coevolutionary genetic programming* (CGP) approach to reduce the feature dimensionality. They generate a low-dimensional synthesized feature vector from the original high-dimensional feature vector such that the data has Gaussian distribution in the low dimension. As compared to the above methods, CGP does not assume any class distribution in the original visual feature space and yields explicit transformation for dimensionality reduction so that the images can be searched in the low-dimensional feature space.

In our transductive learning scheme presented in this paper, we propose to develop a *coevolutionary feature synthesized EM* algorithm, called CFS-EM. It overcomes several limitations of EM algorithm: (a) "Curse of dimensionality," which means that in high-dimensional feature space it needs a large amount of data and the computational cost varies exponentially with the dimensionality. (b) As a hill-climbing kind of method, the convergence of EM algorithm is guaranteed only at a local maximum. The CFS-EM is a hybrid of *coevolutionary genetic programming* (CGP) and EM algorithm applied on partially labeled data. The EM algorithm can help refine the associated classifier from CGP to a local maximum and CGP has a high likelihood that it will jump out of a local maximum to provide near optimal results and a better estimation of parameters. CFS-EM synthesizes low-dimensional features based on CGP algorithm, which yields near optimal non-linear classifier. The usage of small amount of labeled data in CFS-EM is more practical from both the considerations of the expensive labeling of large amount of images and the subjectivity of human image perception. Further, the unlabeled data are boosted with the help of the class distribution learning using the CGP feature synthesis approach. Experiments on real image databases show that CFS-EM outperforms RBF-SVM, CGP, *Discriminant-EM* (D-EM), and *Transductive SVM* (TSVM) in the sense of classification performance and it is more efficient than RBF-SVM in the query phase.

The rest of this article is organized as follows. Section 2 provides the related work, and Section 3 gives the contribution of this article. Section 4 describes the image retrieval system and the CFS-EM algorithm. Section 5 presents extensive comparisons between CFS-EM, RBF-SVM, CGP, D-EM and TSVM. Finally, Section 6 concludes the article.

2. RELATED WORK

Starting with the first CBIR system *QBIC* [Flickner et al. 1995], there has been an increasing demand for management and efficient retrieval of pictorial data [Smeulders et al. 2000]. Since then a number of systems have emerged. They include *BlobWorld* [Carson et al. 2002] working with segmented "blobs" with texture and color features and 3*D MARS* [Nakazato and Huang 2001] providing visualization of results. Other recent systems are: *SIMPLIcity* [Wang et al. 2001] (region-based semantic sensitive queries), I-Browse [Tang et al. 2003] (image and text queries), Netview [Zhu and

Zhang 2000] (multiexample queries) and mental retrieval system [Fang and Geman 2005] (without a query).

The performance of all these systems depends on how the features are selected. As a matter of fact, choosing proper, concise and meaningful features is not trivial. The problem is widely known as *feature selection*. It is defined as selecting a subset of features or constructing new features that are useful to build a good predictor for classification [Guyon and Elisseeff 2003]. In CBIR, a variety of interesting feature selection approaches has been proposed to deal with high dimensional features. [Swets and Weng 1999] propose a self-organizing hierarchical optimal subspace learning and inference framework (SHOSLIF) for image retrieval. The main idea is to recursively implement linear discriminant analysis on the data subsets obtained from the recursive subdivision of the data, so that the limitations of the global linear transformation are overcome. Such a hierarchical linear analysis is a nonlinear approach in nature. Peng et al. [1999] use relevance feedback to find the weights of features. Su et al. [2001] exploit *principal component analysis* (PCA) for dimensionality reduction by using relevance feedback. Wu et al. [2000] reduce the feature dimensionality for image databases using the approach of *weighted multidimensional scaling* (WMDS), whose main characteristic is to preserve the local topology of the high dimensional space. Lee and Seung [1999] use a *nonnegative matrix factorization* (NMF) approach that yields a part-based representation instead of a holistic representation. It is found to be useful for face recognition [Li et al. 2001] and document retrieval [Xu et al. 2003]. He et al. [2003, 2004] propose the approach of *locality preserving projections* (LPP) for face analysis and image retrieval. The advantage of LPP is that it preserves local information by detecting a nonlinear manifold structure. In Yu and Tian [2006], manifolds are learned by semantic subspace projection and one semantic concept may correspond to multiple subspaces. Genetic algorithms and genetic programming have also been used for feature selection for object recognition [Bhanu and Lin 2003] and CBIR [Dong et al. 2005].

However, the reduction in dimensionality of labeled data and supervised learning are not sufficient for content-based image retrieval. It needs to be combined with unsupervised learning (EM algorithm) for effective image retrieval and to overcome the barriers of both the high dimensionality and the lack of labeled data, another important problem in CBIR.

Manual labeling of each image is expensive and subjective. So researchers have come with the idea of *semi-supervised* learning. One approach is to use the users' feedback, called *relevance feedback* [Rui et al. 1998; Peng et al. 1999; Yin et al. 2002; Dong and Bhanu 2003, 2005; Yin et al. 2005] to increase the size of labeled data. But it needs human feedback, which is not always available. Another main approach (discussed here) is called *transductive learning* [Virginia 1993]. It combines both labeled and unlabeled data in training. In this scheme, labeled data provide the initialization and validation of the classifier, and the unlabeled data capture the statistical characteristics of the dataset and boosts the classifier. Various transductive learning methods have emerged during the past 10 years. Joachims [1999] uses SVM scheme to maximize the margin for both the unlabeled data and the labeled data by assigning the unlabeled data to proper classes. Chen et al. [2003] propose an improvement of Joachims's work. This method can handle the cases when the training sets are small or when there is a significant difference in distributions of the labeled and unlabeled data. In the above two approaches, all the unlabeled data are given equal confidence on their prediction. So in another version of transductive SVM [Saunders et al. 1999], instead of using margin transduction, it defines "confidence" and "credibility" for each unlabeled sample and gets the best prediction for the unlabeled data by maximizing the total credibility. Besides SVM, other learning methods are also used in the transductive learning scheme. Bargeron et al. [2005] propose a boosting-based transductive learning. The idea is to construct a diversity of unsupervised models of unlabeled data using a clustering algorithm. These models are then exploited to construct a number of hypotheses using the labeled data and the learner selects a hypothesis that minimizes a transductive error bound. Joachims [2003] proposes a transductive version of K nearest

neighbor classifier, in which the training problem has a meaningful relaxation that can be solved globally optimally using a spectral method. Wu et al. [2000] use a feature selection + EM scheme to solve the transductive learning problem. Transductive learning has been applied in text/Web page classification [Joachims 1999; Nigam et al. 2000; Joachims 2003; Bargeron et al. 2005], color/texture segmentation [Wu and Huang 2000; Xiang et al. 2005], image retrieval [Wu et al. 2000], text detection [Bargeron et al. 2005] and digit recognition [Saunders et al. 1999]. As we can see, most applications of transductive learning are on text classification, and not much work has been done on image retrieval. So in our CFS-EM work, we attempt to contribute to this field.

Our method combines feature synthesis and clustering in a transductive learning scheme. It is a new approach to reduce the feature dimensionality and to learn the classifier in the lower dimension, which reduces the requirement of large amount of labeled training data for high dimensional feature space and improves the query efficiency. The large amount of unlabeled data which characterize the joint probability distribution across different features are used to boost the weak classifiers so that the labeled data and the classified unlabeled data work together to explore discriminating features in a self-supervised fashion to improve the classifier learning.

3. CONTRIBUTIONS OF THIS ARTICLE

As compared to the state of the art already outlined, the contributions of this article are: 1) a new algorithm is presented that combines *coevolutionary feature synthesis* (CFS) with the *Expectation-Maximization* (EM) algorithm on partly unlabeled data, such that both the high feature dimensionality problem and the labeling problem are solved; 2) The CGP algorithm is used to reduce the feature dimensionality, so an explicit transformation is obtained to expedite the query processing; 3) EM algorithm is used to refine the learned classifier from CGP at each iteration. As a hill-climbing algorithm, EM converges to one of the local maxima, which may not be the global maximum. Although not guaranteed, CGP has a high likelihood to jump out of a local maximum within EM iterations and may potentially allow it to achieve the global maximum; 4) The experimental results on various datasets are presented and a comprehensive comparison of different approaches is presented to demonstrate the efficacy and efficiency of the proposed approach.

To help the readers better understand the paper, we show the definition of all the key symbols in this paper in Table I.

4. CFS-EM IMAGE RETRIEVAL SYSTEM

The CFS-EM image retrieval system diagram is shown in Figure 1. In the training phase, the inputs are labeled (L) and unlabeled (U) data. By applying the CFS-EM algorithm, the best feature synthesis transformation (composite operator vector) and a Bayesian classifier in low dimension (two blocks in shade) are obtained. During the testing phase, a query image is transformed into a synthesized feature vector in the low dimension by applying the learned transformation. Then this synthesized feature vector is classified into a known class using the Bayesian classifier. Within this class, Euclidean distance is calculated between the query and every class member. Those images with shortest distances are regarded as most similar to the query and returned as the query result. The key components of CFS-EM algorithm (coevolutionary feature synthesis and EM algorithm) and the algorithm itself will be explained in the following subsections.

4.1 Coevolutionary Feature Synthesis Based on CGP

Coevolutionary feature synthesis is implemented with a *coevolutionary genetic programming* (CGP) approach [Koza 1994; Bhanu et al. 2005; Lin and Bhanu 2005]. It creates low dimensional synthesized feature vectors from the original high dimensional feature vectors, also called *primitive features*.

Table I. Definition of the Symbols Used in This Paper

C	The number of classes	β_i, γ, t	The parameters for RBF-SVM classifier
D	The original feature dimensionality	$\alpha_i, \mathbf{u}_i, \Sigma_i$	Parameters of Gaussian mixture (component weights, means and covariance matrices)
d	The reduced feature dimensionality		
U	The unlabeled training data	θ	The whole parameter set for Gaussian mixture ($\alpha_i + \mathbf{u}_i + \Sigma_i$ in EM algorithm) or the Bayesian classifier in CFS-EM and CGP.
L	The labeled training data		
λ	The percentage of support vectors in L	POP	The whole population in CGP & CFS-EM
T	The total number of training data $\|L+U\|$	$CO(.)$	Composite operator vector
l	L in low dimension	P_i	Sub-population in CGP
u	U in low dimension	G	The number of generations in CGP
r	The percentage of $\|L\|$ in T, $r = \|L\|/\|L+U\|$	M	The size of subpopulation in CGP
$\mathcal{X}$	The dataset in EM algorithm	S	The number of subpopulation (the number of composite operators, the number of synthesized features) in CGP
$\mathcal{Y}$	The labels for $\mathcal{X}$ in EM algorithm and the labels for $L+U$ in CFS-EM algorithm		
W	$L, U+\mathcal{Y}$.	K	The maximum composite operator size

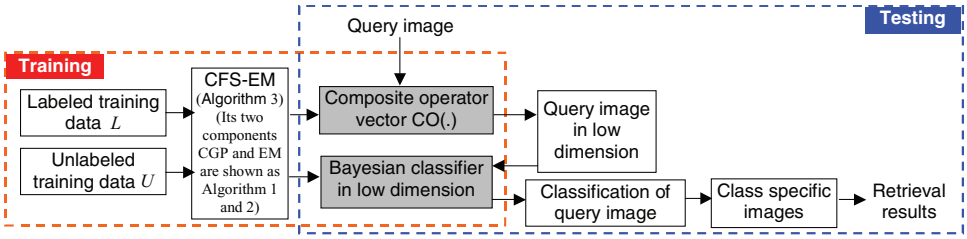


Fig. 1. Coevolutionary feature synthesized—EM (CFS-EM) classification system for image retrieval.

The synthesized features are obtained by applying a series of operators to the original visual features. Such operators are called *composite operators*, which will be explained in Section 4.1.1. These composite operators map the original feature space to the low-dimensional synthesized feature space, in which the images belonging to the same class form a Gaussian component no matter how these images are distributed in the original visual space. Theoretically, this makes EM in low dimension possible.

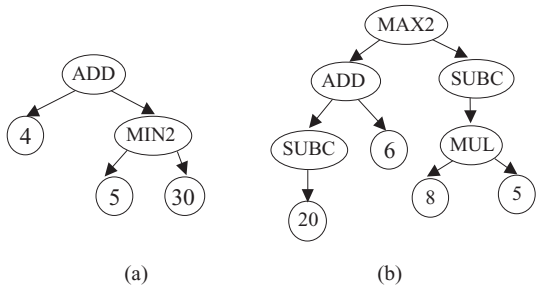


Fig. 2. Sample composite operators: leaf nodes (labeled as 4, 5, 6 etc.) are original visual features and inner nodes (ADD, MAX2, SUBC, etc.) are primitive operators. (a) A simple composite operator (b) A complex composite operator.

Table II. Twelve Primitive Operators

Primitive Operator	Description	Primitive Operator	Description
ADD (a, b)	Add a and b.	ADDC (a, c)	Add constant value c to a.
SUB (a, b)	Subtract b from a.	SUBC (a, c)	Subtract constant value c from a.
MUL (a, b)	Multiply a and b.	MULC(a, c)	Multiply a with constant value c.
DIV (a, b)	Divide a by b.	DIVC (a, c)	Divide a by constant value c.
MAX2 (a, b)	Get the larger of a and b.	MIN2 (a, b)	Get the smaller of a and b.
SQRT (a)	Return sqrt(a) if a $\geq$ 0; otherwise, return –sqrt(-a).	LOG (a)	Return log(a) if a $\geq$ 0; otherwise, return – log(-a).

The search space of all possible composite operators is so large that it is extremely difficult to find good composite operators from this vast space unless a smart search strategy is used. We follow the CGP algorithm proposed in Lin and Bhanu [2005], which was designed for object recognition. During training, CGP runs on labeled training images and evolves to get the best set of composite operators and a Bayesian classifier in the synthesized feature space. During testing, a composite feature vector (low-dimensional vector) is first obtained by applying the composite operator vector on the primitive features of the testing image, and then the Bayesian classifier is used for the classification.

4.1.1 *Composite Operators (Binary Tree).* The composite operators are represented by binary trees with primitive features as the leaf nodes and primitive operators (simple math operations: ADD, SUB, etc.) as the internal nodes. Primitive operators will be explained in the next subsection. Each tree corresponds to one composite operator, which operates on the primitive features and gives a composite feature at the root node. A set of composite operators is called a composite operator vector. Figure 2 gives two sample composite operators, which are real examples from our experiments. Note that not all the original features are necessarily used. The number of composite operators equals to the number of subpopulations.

4.1.2 *Primitive Operator Set.* A primitive operator is an inner node of the composite operator that takes one or two real numbers, performs a simple operation on them and outputs the result. Table II shows the 12 primitive operators that we use, where a and b are real numbers as input to an operator and c is a constant real number stored in an operator.

4.1.3 *Fitness Measure.* During training, the fitness of a composite operator vector is computed in the following way. First, we apply each composite operator of the composite operator vector on the original feature vectors of training images to obtain composite feature vectors. Then, we use these

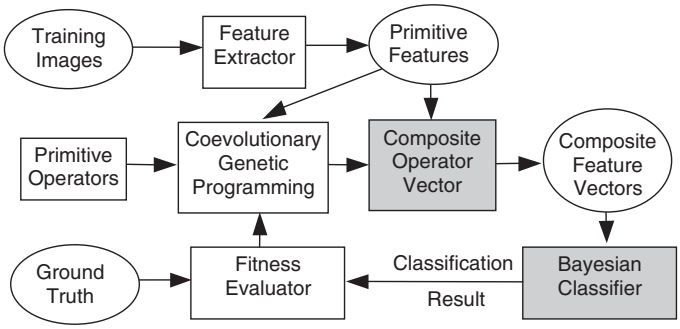


Fig. 3. CGP Training module: learning of the composite operator vector and the Bayesian classifier.

vectors to design a Bayesian classifier. The classification accuracy on the training data is the fitness of the composite operator vector. Figure 3 shows the training procedure.

4.1.4 *Generational Coevolutionary Genetic Programming (CGP).* Generational coevolutionary genetic programming is shown as in Algorithm 1.

Algorithm 1. Generational coevolutionary genetic programming (CGP)

Input: *primitive feature vectors with ground truth labels, number of composite operators (the number of subpopulations)*
Output: *composite operator vector, Bayesian classifier in low dimension.*
Begin:
1. Randomly generate S sub-populations of size M and evaluate each composite operator in each sub-population individually.
2. **FOR** $gen = 1$ to G **DO**
   FOR $i = 1$ to S **DO**
   1) Keep the best composite operator in sub-population P_i .
   2) Perform crossover on the composite operators in P_i until the crossover rate is satisfied and keep all the offspring from crossover.
   3) Perform mutation on the composite operators in P_i and the offspring from crossover with the probability of mutation rate.
   4) Perform selection on P_i to select some composite operators and combine them with the composite operators from crossover and mutation to get a new sub-population $P_i^{'}$ of the same size as P_i.
   5) Evaluate each composite operator CO_j , $j = 1, \ldots, M$ in $P_i^{'}$.
   6) Perform elitism replacement.
   7) Form the current best composite operator vector consisting of the best composite operators from corresponding sub-populations and evaluate it. If its fitness is above the fitness threshold, goto 3.
   END FOR
  END FOR
3. Select the best composite operator from each sub-population to form the learned composite operator vector and output it with the corresponding Bayesian classifier.
End

This algorithm evolves composite operators. The composite operators (binary trees) in the initial subpopulations are randomly generated. A composite operator is generated in two steps. In the first step, the number of internal nodes of the tree representing the composite operator is randomly determined as long as this number is smaller than half of the *maximum operator size* (K). The tree is generated from top to bottom by a tree generation algorithm. The root node is generated first and the primitive

operator stored in the root node is randomly selected. The selected primitive operator determines the number of children the root node has. Suppose the tree has p internal nodes. If it has only one child, the algorithm is recursively invoked to generate a tree of p–1 internal nodes; if it has two children, the algorithm is recursively invoked to generate two trees of (p–1)/2 and (p–1)/2 internal nodes, respectively. In the second step, after all the internal nodes are generated, the original features are randomly picked as the leaf nodes. The GP operations are applied in the order of crossover, mutation, and selection. In addition, an elitism replacement method is adopted to keep the best composite operator from generation to generation.

To evaluate the jth composite operator CO_j in step 5) in Algorithm 1, the current best composite operator in each of the other subpopulations is selected and combined with CO_j to form a composite operator vector where composite operator from the jth subpopulation occupies the jth position in the vector ($j = 1, \ldots, S$). Note that the reduced feature dimensionality is the same as the number of subpopulations (S). The composite operator vector is run on the original features of the training images to get composite feature vectors, and they are used to build a Bayesian classifier. The classification accuracy of the Bayesian classifier is the fitness of the composite operator vector and CO_j.

4.1.5 *Parameters and Termination.* The key parameters are the number of subpopulations (the number of synthesized features, the number of composite operators) S, the size of subpopulation M, the number of generations G, the crossover and mutation rates, tournament size, and the fitness threshold. CGP stops whenever it finishes the specified number of generations or the performance of the Bayesian classifier is above the fitness threshold. After termination, CGP selects the best composite operator of each subpopulation to form the learned composite operator vector and the Bayesian classifier to be used during the testing.

The number of subpopulation, and the size of subpopulation are dataset-dependent parameters and they are related with the size of the search space. Normally they are a very small percentage of the size of the search space [Bhanu and Lee 1994; Bhanu et al. 2005]. On one hand, the higher the number and size of subpopulations are, the higher the computational cost (during both the training and testing phases) is. On the other hand, a smaller number and size of subpopulations will require more generations to explore the search space. The previous research has shown that at the two extremes of subpopulation number (a very large number or only one), the performance is likely to be slightly lower [Lin and Bhanu 2005]. So, we have determined these two parameters and the number of generations based on empirical experiments from our previous work [Dong et al. 2005; Li et al. 2005]. Other parameters (crossover rate, mutation rate and tournament size) are well explored in the literature on evolutionary computation, so they are chosen based on the experience [Koza 1994; Bhanu et al. 2005].

4.1.6 *Selection, Crossover, and Mutation.* The CGP searches through the space of composite operator vectors to generate new composite operator vectors. The search is performed by selection, crossover and mutation operations. The initial subpopulations are randomly generated. Although subpopulations are cooperatively evolved (the fitness of a composite operator in a subpopulation is affected by the composite operators from other subpopulations), selection, crossover and mutation are performed separately within each subpopulation.

—*Selection.* This operation involves selecting composite operators from the current sub-population. We use tournament selection[1] [Koza 1994], and the tournament size is set as 5. The higher the fitness value is, the more likely the composite operator is selected.

[1]A group of individuals is randomly selected and the fittest one of this group is picked for crossover and mutation. This scheme is simulating the biological mating pattern in which several members of the same sex compete to mate with another one with a different sex. The size of the group is tournament size (usually 2–7).

—*Crossover.* Two composite operators, called parents, are selected on the basis of their fitness values. The higher the fitness value is, the more likely that a composite operator is selected as one of the parents for crossover. One internal node in each of these two parents is randomly selected, and the two subtrees rooted at these two nodes are exchanged between the parents to generate two new composite operators, called offspring. It is easy to see that the size of one offspring (i.e., the number of nodes in the binary tree representing the offspring) may be greater than both parents if crossover is implemented in such a simple way. To prevent this *code bloat*, we specify the maximum size of a composite operator (called max-operator-size). If the size of one offspring exceeds the max-operator-size, the crossover is performed again. If the size of an offspring still exceeds the max-operator-size after the crossover is performed 10 times, GP selects two subtrees of the same size (i.e., the same number of nodes) from two parents and swaps the subtrees between the parents. These two subtrees can always be found, since a leaf node can be viewed as a subtree of size 1.

—*Mutation.* To avoid premature convergence, mutation is introduced to randomly change the structure of some composite operators to maintain the diversity of subpopulations. Candidates for mutation are randomly selected and the mutated composite operators replace the old ones in the subpopulations. There are three mutations invoked with equal probability: 1) Randomly select a node of the composite operator and replace the subtree rooted at this node by a randomly generated binary tree. 2) Randomly select a node of the composite operator and replace its primitive operator with another primitive operator randomly selected from the primitive operators of the same number of input as the replaced one. 3) Randomly select two subtrees of the composite operator and swap them. Of course, neither of the two subtrees can be a subtree of the other.

4.2 Expectation-Maximization (EM)

We assume feature vectors in the low dimension follow a C component *Gaussian Mixture Model* (GMM) [McLachlan and Peel 2000]. They can be regarded as samples of a d-dimensional random variable X, where $\mathrm{x} = [x_1, \ldots \mathrm{x}_d]^T$ represents a particular sample. Its probability density function is defined as $p(\mathbf{X}|\theta) = \Sigma_{i=1}^{C} \alpha_i f_i(\mathbf{x}|\theta_i) = \Sigma_{i=1}^{C} \alpha_i f_i(\mathbf{x}|\mathbf{u}_i, \Sigma_i)$. In this definition, $\alpha_1, \ldots \alpha_C$ are the *mixture probabilities*, so they are positive and sum up to 1. θ_i is the set of parameters for the ith Gaussian component, which includes mean μ_i and covariance matrix Σ_i. Thus, $\theta \equiv \{\alpha_1, \ldots \alpha_C, \theta_1, \ldots \theta_C\}$ is the complete set of parameters needed to specify the mixture. Given a set of N independent samples of $\mathbf{X}$: $\mathcal{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$, the goal is to find θ which maximizes $\log \mathrm{p}(\mathcal{X}|\theta)$ (maximum likelihood, ML) or $\log \mathrm{p}(\mathcal{X}|\theta) + \log \mathrm{p}(\theta)$ (maximum a posteriori, MAP). *Expectation Maximization* (EM) is such an algorithm [Dempster et al. 1977] to find θ. It is based on the interpretation of $\mathcal{X}$ as incomplete data. The missing part is a set of N labels $\mathcal{Y} = [\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)}]$ associated with the N samples, indicating which component produces each sample. $\mathbf{y}^{(n)} = [y_1^{(n)}, \ldots, y_C^{(n)}]$, where $y_k^{(n)} = 1$ and $y_j^{(n)} = 0$ for $j \neq k$, means that sample $\mathbf{y}^{(n)}$ is produced by the kth component. The complete log-likelihood is shown in Equation (1). The EM algorithm is described in Algorithm 2. It produces a sequence of estimates $\{\hat{\theta}(t), t = 0, 1, 2, \ldots\}$ by alternatively applying the following E-step and M-step until some convergence criterion is met.

$$\log p(\mathcal{X}, \mathcal{Y}|\theta) = \sum_{n=1}^{N} \sum_{i=1}^{C} y_i^{(n)} \log \left[\alpha_i p\left(\mathbf{x}^{(n)}|\theta_i\right)\right]. \tag{1}$$

—*E-step.* Compute the conditional expectation of the complete log-likelihood, given $\mathcal{X}$ and the current estimate $\hat{\theta}(t)$. The result is the so-called Q-function:

$$Q(\theta, \hat{\theta}(t)) \equiv E[\log p(\mathcal{X}, \mathcal{Y}|\theta)|\mathcal{X}, \hat{\theta}(t)] = \log p(\mathcal{X}, \mathcal{Z}|\theta). \tag{2}$$

In this equation, because of the linearity of $\log p(\mathcal{X}, \mathcal{Y}|\theta)$ with respect to $\mathcal{Y}$, we only need to compute the conditional expectation $\mathcal{Z} \equiv E[\mathcal{Y}|\mathcal{X}, \hat{\theta}(t)]$. It is explicitly given by Equation (3), where $z_i^{(n)}$is a posteriori probability that $y_i^{(n)}$ =1, after observing $\mathbf{x}^{(n)}$.

—*M-step.* Update the parameter estimates according to $\hat{\theta}(t+1) = \arg\max_{\theta} Q(\theta, \hat{\theta}(t))$ in the case of ML estimation, or $\hat{\theta}(t+1) = \arg\max_{\theta}\{Q(\theta, \hat{\theta}(t)) + \log p(\theta)\}$ in the case of MAP estimation. In our approach, ML estimation is used.

$$z_i^{(n)} \equiv E\left[y_i^{(n)}|\mathcal{X}, \hat{\theta}(t)\right] = \Pr\left[y_i^{(n)} = 1|\mathbf{x}^{(n)}, \hat{\theta}(t)\right] = \frac{\hat{\alpha}_i(t)p(\mathbf{x}^{(n)}|\hat{\theta}_i(t))}{\sum_{j=1}^{C}\hat{\alpha}_j(t)p(\mathbf{x}^{(n)}|\hat{\theta}_j(t))}. \quad (3)$$

To apply the EM algorithm in image retrieval, Cin equation (1) denotes the number of classes, which is given as a *priori*. C can be determined by following the algorithm in Figueriedo and Jain [2002].

Algorithm 2. EM algorithm for finite mixture model learning

Input: *samples* $\mathcal{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$**,** ε(*evaluate the difference of Q functions*)
Output: *Mixture model in* $\hat{\theta}_{best}$
Begin:
Initialize $\hat{\alpha}_1 = \cdots = \hat{\alpha}_C = 1/C$. Randomly pick C samples as the means $\{\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_C\}$. Get the covariance matrix of $\mathcal{X}$ as the covariance matrix of each class $\{\hat{\Sigma}_1, \ldots, \hat{\Sigma}_C\}$.
$u_i^{(n)} \leftarrow p(\mathbf{x}^{(n)}|\hat{\theta}_i)$,for $i = 1, \ldots, C$, and $n = 1, \ldots, N$.
repeat
$t \leftarrow t + 1$.
FOR i = 1 to C **DO**
$z_i^{(n)} \leftarrow \hat{\alpha}_i u_i^{(n)}(\sum_{j=1}^{C}\hat{\alpha}_j u_j^{(n)})^{-1}$, for n =1, ... , N. (equation (3))
$\hat{\alpha}_i = \sum_{n=1}^{N} z_i^{(n)}/N$.
$\hat{\theta}_i \leftarrow \arg\max_{\theta_i} \log p(\mathrm{X}, \mathrm{Z}|\theta)$. (equation (2))
$u_i^{(n)} \leftarrow p(\mathbf{x}^{(n)}|\hat{\theta}_i)$,for n= 1, ... , N.
END FOR
$\hat{\theta}(t) \leftarrow \{\hat{\theta}_1, \ldots, \hat{\theta}_C, \hat{\alpha}_1, \ldots, \hat{\alpha}_C\}$, $Q(\mathrm{t}) = Q(\theta, \hat{\theta}(t))$.
until $|Q(\mathrm{t}) - Q(\mathrm{t}-1)| < \varepsilon$
$\hat{\theta}_{best}$ $\hat{\theta}(t)$.
End

4.3 Coevolutionary Feature Synthesized-EM (CFS-EM) Algorithm

In this work, we combine CGP with EM, to overcome the high dimensional visual feature classification task by integrating supervised and unsupervised learning paradigms and identifying most discriminating features in a self-supervised fashion. The CFS-EM algorithm is described in Algorithm 3. In the initialization, CGP is applied on labeled training data (L) to get a composite operator vector CO($\cdot$) and a Bayesian classifier represented by the Gaussian distribution parameters θ. The population corresponding to the best composite operator vector is saved (POP). Both labeled training data (L) and unlabeled training data (U) are transformed into low dimension (l and u) using this composite operator vector CO($\cdot$). In CFS-EM iteration, first *EM hill climbing* is applied on this low dimension dataset ($l + u$) to find a locally optimal Bayesian classifier. The stopping criterion for EM is that the parameters (θ) keep unchanged for two consecutive iterations. At this time, both L and U are "labeled" by the Bayesian classifier. Labeled training data Lcan be assigned with wrong labels in this step, but they will be changed

to their ground-truth label (step 2). In the *Jump out of local maximum* step, CGP is applied on the "labeled" whole dataset to find a better composite operator vector (step 6) and update the Bayesian classifier. The saved population is used as the initial population for CGP and it is updated by the new population to keep the performance increasing. This is an improvement from our previous work [Li et al. 2005], where the population of the next iteration is generated randomly. The *Hill climbing* and *Jump out of local maximum* steps iterate until one of the three criteria is reached: 1) a certain number of iterations is run, 2) a satisfactory classification performance is reached, or 3) the fitness value does not change for 5 iterations.

Algorithm 3. Coevolutionary Feature Synthesized – EM (CFS-EM)

Input: *labeled training dataset L from C classes in original feature space*
Unlabeled training dataset U in original feature space
Synthesized feature dimension d
CGP parameters (sub-population size, crossover rate, mutation rate, maximum composite operator size, fitness value, tournament size and generation number)
Output: *Composite operator vector CO(·)*
Bayesian classifier in low dimension θ
Begin:
Initialization:
$[\mathrm{CO}(\cdot), \theta] = \mathrm{CGP}(L)$; initialize randomly and save the population corresponding to the best $\mathrm{CO}(\cdot)$ as POP.
$l = \mathrm{CO}(L)$, $u = \mathrm{CO}(U)$.
CFS-EM iteration:
Hill climbing:
1) Get labels ($\mathcal{Y}$) for $l + u$ (by calculating $\mathcal{Z}$ based on equation (3)).
2) Change the labels ($\mathcal{Y}$) of l to the ground-truth labels.
3) Update θ based on $\mathcal{Y}$.
4) If θ does not change much, goto 5), otherwise, goto 1).
Jump out of local maximum:
5) $W = L, U + \mathcal{Y}$. $//W$ is the "labeled" whole training dataset (L, U together with their labels $\mathcal{Y}$)
6) $[\mathrm{CO}(\cdot),\theta] = \mathrm{CGP}(W)$ (using POP as the initial population and update POP at the end).
7) $l = \mathrm{CO}(L)$, $u = \mathrm{CO}(U)$, goto 1).
End

The advantages of CFS-EM are: 1) The unlabeled data can be boosted with the help of the class distribution learning using CGP feature synthesis approach; 2) It synthesizes low-dimensional features based on *coevolutionary genetic programming* (CGP) algorithm, which yields near "optimal" nonlinear transform; 3) The explicitness of feature transformation is especially suitable for image retrieval because the images can be searched in the synthesized low-dimensional space, while kernel-based methods have to make classification computation in the original high-dimensional; 4) The possibility exists of helping EM to jump out of local maximum and reach the global maximum.

5. EXPERIMENTAL RESULTS

To evaluate the efficacy of CFS-EM approach, we evaluate its classification performance and query efficiency on five real databases. We also compare it with RBF-SVM, CGP, D-EM [Wu et al. 2000], and TSVM [Joachims 1999] approaches.

Fig. 4. Sample image of the 12 classes in *Corel-1200* obtained from the Corel stock photo library.

5.1 Datasets

—*Corel-1200.* We select 1200 images belonging to 12 classesfrom Corel Stock photo library [Dong and Bhanu 2003] (*www.corel.com*), and the images in each class have similar visual features. We assume that the images in each class form a Gaussian cluster in the feature space. Figure 4 shows the sample image from each class.

—*Corel-1500.* We add 300 images (from three other CDs in the library) into *Corel-1200* to obtain *Corel-1500*. The three new CDs are *hawks and falcons*, *tigers*, and *tulips*. They are merged to *owl, lions*, and *North American wildflowers* in *Corel-1200,* respectively. Thus, *Corel-1500* still has 12 classes [Dong and Bhanu 2003]. The purpose of *Corel-1500* is to demonstrate that CGP can map the original features to a low-dimensional space and make their distribution a Gaussian no matter how they are distributed originally. Since each class is no longer a Gaussian cluster, this database is challenging for linear transformation D-EM.

—*Corel-6600.* This dataset is also extracted from Corel stock. It contains 50 classes and there are ∼100 images in each class. The total number of images in this dataset is 6600.

—*Corel-10038.* This dataset is obtained from Corel stock. It contains 56 classes and a total of 10,038 images. The same dataset was used by Yin et al. [2005] for relevance feedback experiments. The dataset is quite uneven for different classes. The maximum class has 695 images and the minimum class has only 20 images. The small classes do not have enough samples to fit the Gaussian distributions. So we redistribute samples from the three smallest classes (*obelisk, golf course, and national flag*) into the visually closest classes. In our experiments, this dataset has 53 classes and the minimum class has 60 images. All the other classes have varying number of images: 62, 65, . . . , 395, . . . , 695.

—*Flickr-21200.* This dataset is downloaded from a popular online photo management and sharing website *flickr.com*. In this database, each picture is tagged with some key words by the users to describe its content. An API is provided to search and download images from the website. In our experiments, 53 classes similar to *Corel-10038* are searched by tag and 400 images for each class are downloaded. So, there are a total of 21,200 images in this dataset. As compared to the Corel database, all the images downloaded with the same tag are not semantically coherent. So the performance on this dataset can reflect the performance of the classifier on the real world data better.

For the first three datasets, each image is represented by 16 texture features, 6 color features, and 18 structural features, a total of 40 features. For the other two datasets, only texture features and color features are used (totally 22). The 16 texture features are means and standard deviations derived from 8 Gabor filters (2 scales and 4 orientations). The 6 color features are means and standard deviations of the HSV color channels. We use a water-filling approach [Zhou et al. 1999] to extract the 18 structural features.

5.2 Experimental Settings

All the features in the above five datasets are normalized (to range [0,1]). The whole dataset is split into halves, one half for training and the other half for testing. A percentage of the training data (20%, 40%, 60%, and 80%) is defined as labeled training data (L) and the rest (80%, 60%, 40%, and 20%) as unlabeled training data (U). For a certain percentage of labeled data, the average classification accuracy and precision-recall curves on the testing data are compared among CFS-EM, RBF-SVM, CGP, D-EM, and TSVM. As supervised learning schemes, CGP and RBF-SVM use L alone for training. As transductive learning algorithms, CFS-EM, D-EM, and TSVM use both L and U. CFS-EM, D-EM, and CGP reduce the feature dimensionality from 40 to 6 for *Corel-1200, Corel-1500, and Corel-6600*, and from 22 to 4 for *Corel-10038* and *Flickr-21200*. Because of the unbalanced class sizes in *Corel-10038* even after redistributing the samples from the smallest classes, some small classes (minimum class size is 6 for the training data when 20% of training data are labeled) have not enough samples to fit a 6-dimensional Gaussian distribution. So for this dataset, we decrease the features dimensionality to 4. We do the same thing to *Flickr-21200* to make it comparable with *Corel-10038*.

We use LIBSVM [Chang and Lin 2001] to evaluate the performance of RBF-SVM. To handle multiclass problem, it works in a one-against-one + voting mode. We also do a five-fold cross validation to obtain the best set of parameters. In our previous work [Li et al. 2005], we have seen that SVM's performance largely depends on the kernel function. So in this paper, we only use the commonly seen best *kernel radial basis function* (RBF). We use SVM^{light} [Joachims 1999] to evaluate the performance of TSVM. To handle multiclass problem, it also works in a one-against-the-rest + voting mode. And we use the same parameter set (cost and gamma) of RBF-SVM for TSVM.

For CFS-EM, we run the program for 10 times. The maximum vote of the classifications of each test image from the 10 runs is used as the final classification of that image. The CFS-EM parameters are: (a) number of subpopulations: 6 for *Corel-1200, Corel-1500 and Corel 6600*, 4 for *Corel-10038* and *Flickr-21200*; (b) subpopulation size: 100; (c) number of generation: 100; (d) crossover rate: 0.6; (e) mutation rate: $0.1 \rightarrow 0.01$ (*exploration and exploitation* scheme is used to decrease the mutation rate from 0.1 to 0.01 with step size 0.01 in 10 iterations. It is fixed for the 100 generations in one iteration); (f) tournament size: 5; (g) fitness threshold: 1.0. CGP uses the same parameter set as CFS-EM except the mutation rate which is fixed at 0.05.

5.3 Classification Performance Comparisons

Figure 5 shows the classification accuracy comparison of the five classifiers on the five datasets. On all the datasets, the accuracy for all the five classifiers increases as the percentage of labeled data increases. As a semi-supervised learning method, CFS-EM has higher accuracy than the supervised learning methods RBF-SVM and CGP. As a transductive learning algorithm, it outperforms linear transductive learning method D-EM and non-linear transductive learning method TSVM. TSVM has the lowest accuracy because it is designed for binary classification tasks. The one-against-the-rest mode makes the two classes unbalanced and TSVM cannot find the good margin for them. In addition, compared with our previous work [Li et al. 2005], the accuracy of CFS-EM improves about 10%$\sim$20%.

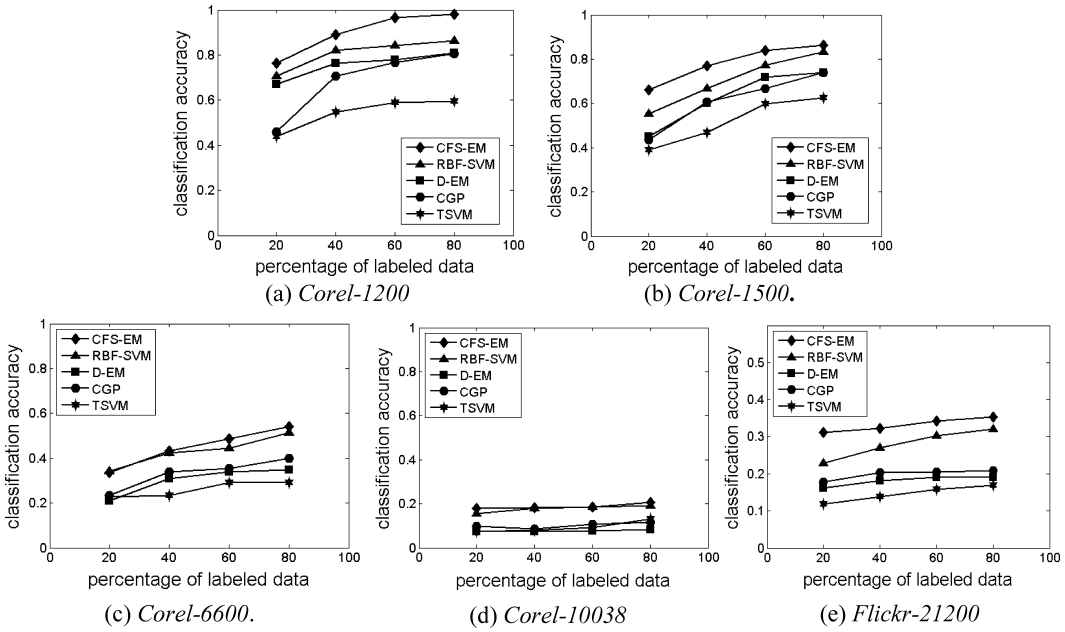
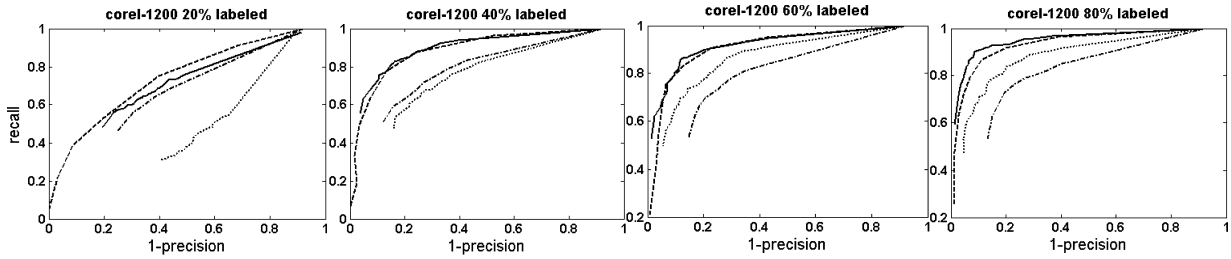


Fig. 5. Classification accuracy comparison on the five datasets.

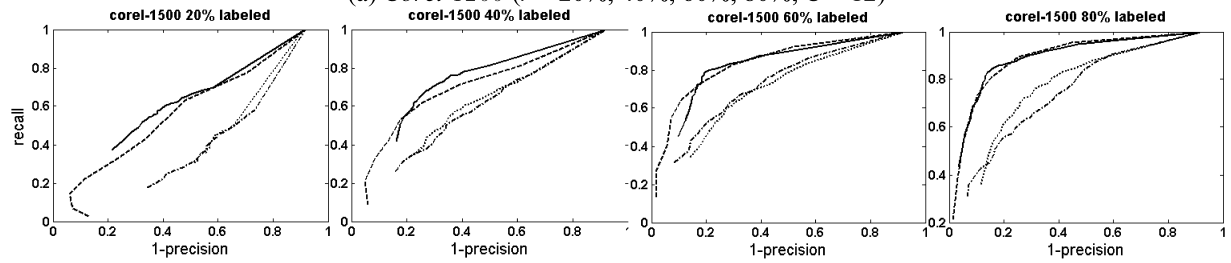
This is because we save the population in the CGP step at an iteration and this population serves as the initial population for the next iteration while the CGP step in the previous approach [Li et al. 2005] starts randomly at each iteration.

Figure 6 shows the precision-recall (PR) curves of CFS-EM, RBF-SVM, D-EM, and CGP on the five datasets at different percentage of the labeled data. Compared with the accuracy curves, a PR curve gives us more detailed information about the tradeoffs. In Figure 6, we can see the same trend as in the accuracy curves: the performance of CFS-EM is higher than RBF-SVM, especially as the number of class increases. CGP and D-EM have lower performance. When comparing CGP and D-EM, it can be seen that for *Corel-1200, Corel-1500, and Corel-6600*, when the percentage of labeled data is low (20%), the performance of D-EM is better than or equal to CGP. This is because D-EM uses another 80% of unlabeled data that CGP cannot use. As the percentage of labeled data increases, the gap decreases and CGP even surpasses D-EM. With the increase in the percentage of labeled data, the advantage of transductive learning is reduced. It justifies the advantages of non-linear transform over linear transform again. For large datasets *Corel-10038 and Flickr-21200,* both D-EM and CGP do not perform well. But we can see that CGP always has higher recall values than D-EM at high precisions, which means it is a better classifier. This explains the higher accuracy for CGP than D-EM in Figure 5 for these two datasets. We do not show the precision-recall plot for TSVM in Figure 6 because it shows the lowest performance in Figure 5.

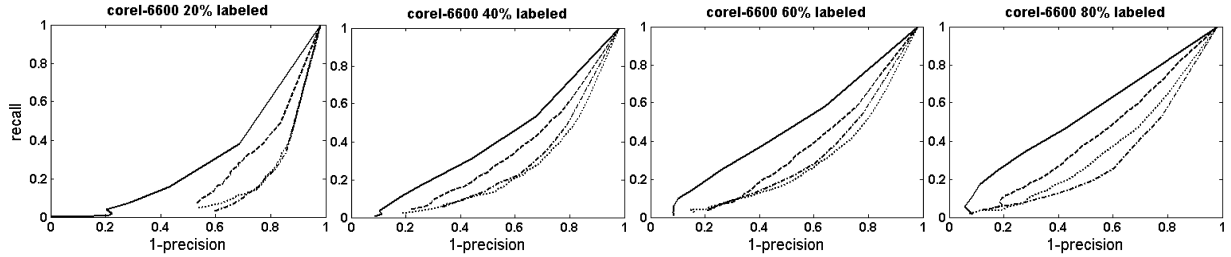
From both Figure 5 and Figure 6, we can see that as the dataset gets larger, the overall classification performance of all the classifiers degrades, and it is especially low for *Corel-10038*. But it is in the same range as the results as shown in a previous work on the same dataset [Yin et al. 2005]. One reason for the low accuracy is that simple features (color, texture) are not good enough to represent an image to discriminate itself from other images in such a large dataset. Another reason is that some ground truth is not correct. For example, Figure 7 shows some sample images from class 7 of *Corel-10038*. They are quite different both visually and semantically to belong to the same class. However, the performance can be



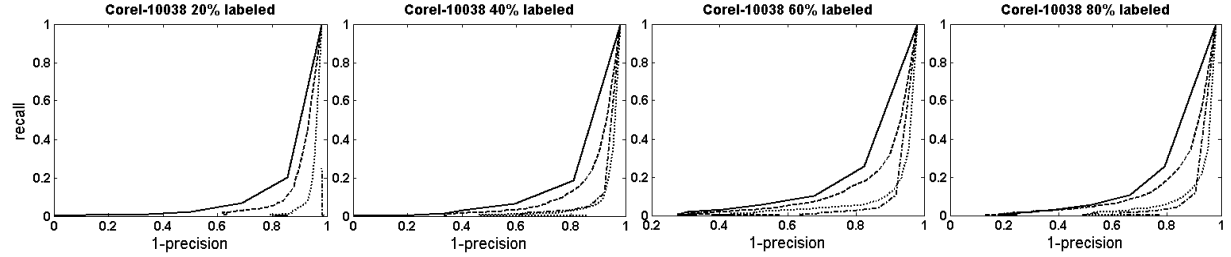
(a) *Corel-1200* (r = 20%, 40%, 60%, 80%, C = 12)



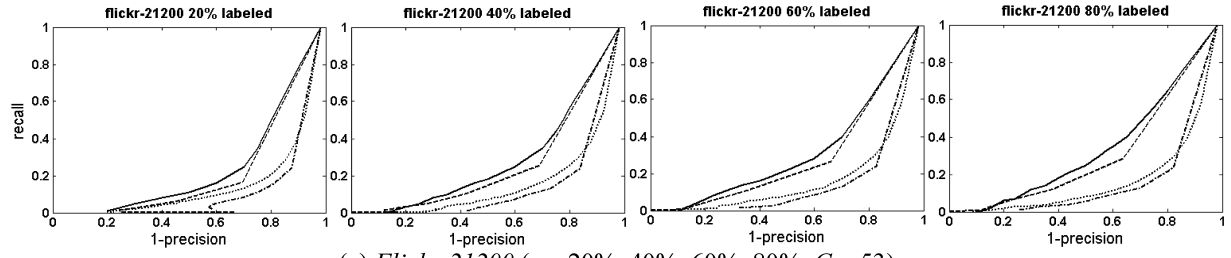
(b) *Corel-1500* (r = 20%, 40%, 60%, 80%, C = 12)



(c) *Corel-6600* (r = 20%, 40%, 60%, 80%, C = 50)



(d) *Corel-10038* (r = 20%, 40%, 60%, 80%, C = 53)



(e) *Flickr-21200* (r = 20%, 40%, 60%, 80%, C = 53)

Fig. 6. Precision-Recall curves of all the classifiers on the five datasets at different percentage of labeled data (Solid line — : CFS-EM, dashed line - - -: RBF-SVM, dotted line . . . : CGP, dash dotted line -.- : D-EM).

Fig. 7. Sample images from class 7 of *Corel-10038* dataset.

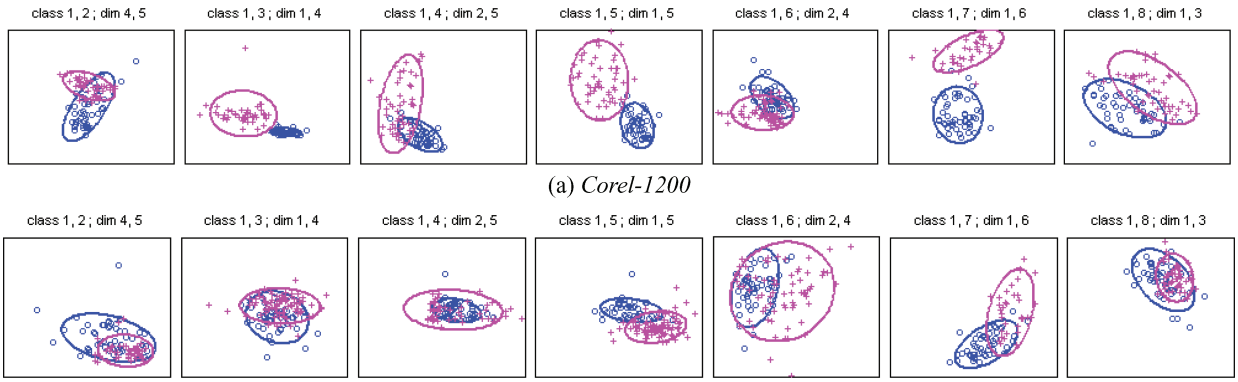
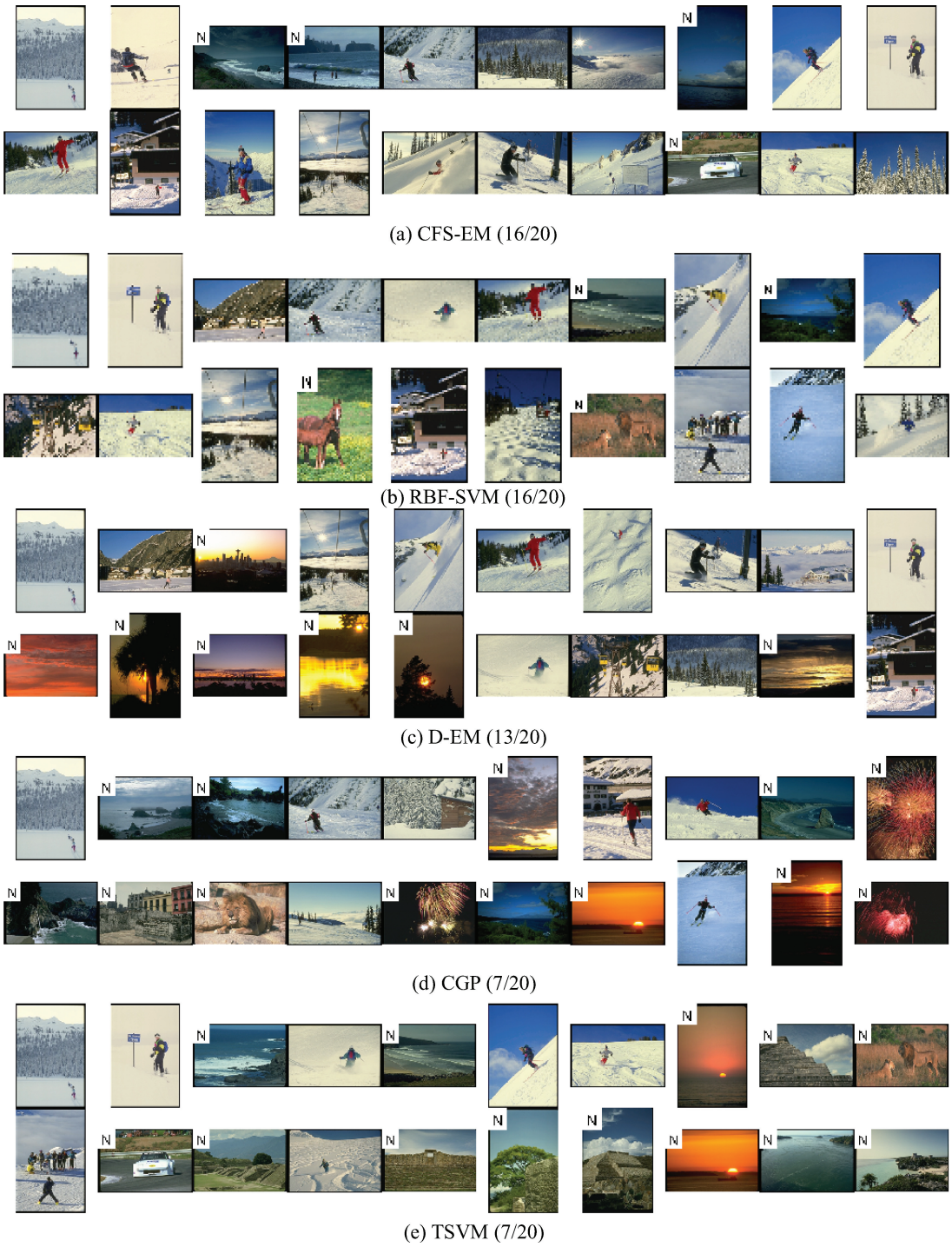


Fig. 8. Projection of synthesized feature vectors of *Corel-1200* and *Corel-1500* on 2D (the two axes are the two projection dimensions which give the best separation for the two classes).

improved by using other techniques like *relevance feedback*. From the existing work on the *Corel-10038* dataset, the overall accuracy could improve from 25.41% to 61.96% by introducing users' feedback [Yin et al. 2005]. With the similar classes as in *Corel-10038*, *Flickr-21200* has a better performance than *Corel-10038*. Even though *Flickr-21200* is also noisy, its classes are balanced. Unbalanced classes make the evolution to favor large classes to obtain a higher fitness value.

Figure 8 shows the 2D projections of the low dimension features (6D) of *Corel-1200* and *Corel-1500*. Since there is not a good way to present the 6D feature vectors on the 2D paper, they are projected to 2D. Every subfigure shows the projection of the samples from two classes (circles and crosses) in two dimensions where they have the best separation. The ellipses are the Gaussian ellipses with two times the standard deviation of each class. Because of the lack of space, we only show some representative combinations. It can be seen that some classes are well separated and some are heavily overlapped (e.g., the leftmost subplot of Figure 8(b)). In general, the overlap is less for *Corel-1200* than for *Corel-1500*. But each class in *Corel-1500* is approximately a Gaussian distribution no matter how the original distribution is, which justifies the advantage of nonlinear transformation CGP over linear transformation MDA.

Figure 9 and Figure 10 show the image retrieval results of the five classifiers (trained when $r = 20\%$ and 60%) on *Corel-1200* dataset to visualize the comparative results. The first image from each query result is the query image. Query results (precisions behind the classifier name in each subfigure) support the previous comparison: the number of correct results of CFS-EM > RBF-SVM > D-EM > CGP > TSVM, and they all increase as the percentage of labeled data increases.



(a) CFS-EM (16/20)

(b) RBF-SVM (16/20)

(c) D-EM (13/20)

(d) CGP (7/20)

(e) TSVM (7/20)

Fig. 9. Top 20 query result images on *Corel-1200* using different classifiers (trained when $r = 20\%$, "N" on a picture means a wrong result), ranked from top to bottom, left to right.

(a) CFS-EM (19/20)

(b) RBF-SVM (16/20)

(c) D-EM (16/20)

(d) CGP (14/20)

(e) TSVM (13/20)

Fig. 10. Top 20 query result images on *Corel-1200* using different classifiers (trained when $r = 60\%$, "N" on a picture means a wrong result), ranked from top to bottom, left to right.

5.4 Computational Cost Comparison in the Query Phase

The query efficiency is a significant concern in CBIR. In this section, we show that CFS-EM outperforms RBF-SVM not only on the classification performance, but also on the query time, both theoretically and experimentally. The parameters that affect query time for CFS-EM and RBF-SVM are: class number C, original feature dimensionality D,reduced feature dimensionality d, the amount of training data T, the percentage of labeled training data r and maximum composite operator size K (see Table I for the definition).

For CFS-EM, the maximum computational cost of one query is given by Equation (4), in clock cycles (shortened as cc). The first term is for composite feature transformation. Given K as the maximum number of nodes in a tree, the maximum number of primitive operators used is $K$when all the primitive operators have only one operand and all the K nodes are used. From all the primitive operators in Table II, assume the most complex computation *sqrt (square root)* is used at each node, which costs 22.3 cc,[2] then d composite operators need a maximum of $22.3Kd$ cc. The second term is the calculation of the Gaussian probability to C classes: $C(d^2 + 26.5)$ cc. To find the nearest class we need to find the largest probability among C classes, which needs C cc. This gives the third term.

$$\underbrace{22.3Kd}_{transformation} + \underbrace{C(d^2+26.5)}_{\substack{Bayesian\\ classification}} + \underbrace{C}_{\substack{find\ the\\ closest\ class}} . \tag{4}$$

For RBF-SVM, the classification of x between two classes is defined by Equation (5), where, $x_i, i = 1, \ldots, nSV$ are the support vectors. $nSV = 2\lambda rT/C$. The norm calculation costs D cc and the exponential costs 26.5 cc. So the total calculation for $f(\mathrm{x})$ is $2\lambda rT/C(D+26.5)$ cc. As mentioned in section 5.2, multiclass RBF-SVM is carried by one-against-one + voting mode, so the total computational cost is giving by Equation (6).

$$f(x) = \mathrm{sgn}\left\{\sum_{i=1}^{nSV} \beta_i \exp(-\gamma \|x_i - x\|^2) + t\right\} \tag{5}$$

$$\underbrace{C(C-1)/2 \cdot 2\lambda rT/C \cdot (D+26.5)}_{one-against-one\ classification\ for\ all\ combinations} + \underbrace{C}_{\substack{find\ the\\ max\ vote}} \tag{6}$$

The percentage of support vectors λ is decided by the class number C, feature dimensionality D and percentage of labeled training data r. Table III shows the real experimental data of λ for the five datasets. From our real data, we use *Least Mean Square Error* (LMSE) estimation to get a linear combination of C, D and r as the estimation of λ. The fitting result is shown in Equation (7). The small MSE validates the linear assumption.

Figure 11 shows the computational cost comparison between CFS-EM and RBF-SVM with reference to the percentage of labeled training data for *Corel-1200*. The dotted line is the maximum cost of CFS-EM and the solid line is the cost of RBF-SVM. It can be seen that for RBF-SVM, the cost increases with the percentage of labeled training data while the cost of CFS-EM does not. The figure shows that even when only 2% of the labeled training data are used, RBF-SVM needs more computation than CFS-EM in the query phase. When examining Equation (7), we can see that when r increases (e.g., from 20% to 40%), the percentage of support vectors λ decreases. However, because the absolute of the coefficient

[2]The computational cost (clock cycle) of the operations used in our calculation (SparcStationII) [Bachmann, O., Effective simplification of CR expressions, http://www.mathematik.uni-kl.de/~zca/Reports_on_ca/11/paper_html/paper.html]:

add/sub	1	mult	1.1	div	3.1	sqrt	22.3	log	20.3	exp	26.5	comp	1

Table III. Percentage of Support Vectors λ from the Experiments (RBF-SVM)

	$r = 20\%$	$r = 40\%$	$r = 60\%$	$r = 80\%$
Corel-1200	0.916	0.82	0.87	0.75
Corel-1500	0.906	0.743	0.744	0.743
Corel-6600	0.99	0.965	0.959	0.958
Corel-10038	0.95	0.949	0.924	0.917
Flickr-21200	0.98	0.975	0.956	0.940

$\lambda = -0.11665 * r + 0.00868657 * C + 0.013075 * D + 0.20231$, and $MSE = 0.0066225$ (7).

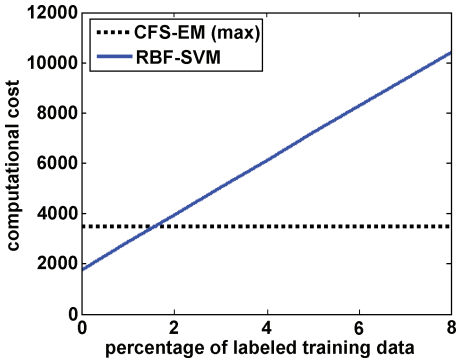


Fig. 11. Computational cost: CFS-EM vs. RBF-SVM. (For *Corel-1200*, $C = 12$, $D = 40$, $d = 6$, $T = 600$, $r = 1\%\sim8\%$, $K = 20$).

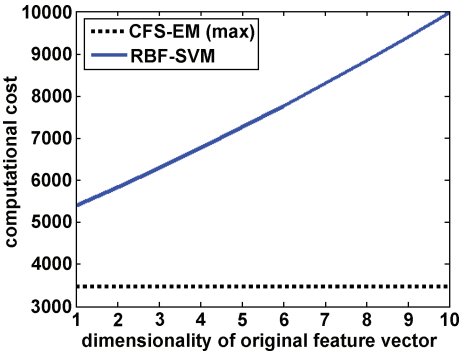


Fig. 12. Computational cost: CFS-EM vs. RBF-SVM. (For *Corel*-1200, $C = 12$, $D = 1\sim10$, $d = 6$, $T = 600$, $r = 20\%$, $K = 20$).

of r is smaller than 1 (it is -0.11665 in Equation (7)), its decrease will not be as fast as the increases in r, so the total number of support vectors still increases. This is why the testing time of RBF-SVM increases when r increases.

From Figure 12 we can see that when the percentage of labeled data is fixed, computational cost of RBF-SVM increases as the number of dimensions of the original feature vector increases while it is fixed for CFS-EM. Even when the dimension of original feature vector is only 1, CFS-EM is faster than RBF-SVM. The reason that CFS-EM is computationally more efficient than RBF-SVM in the query phase is that CFS-EM query works in a low dimensional synthesized feature space while RBF-SVM works in the original high dimensional feature space using lots of the training samples (from Table-III, λ is close to 1, which means almost 100% of the training data are support vectors). In the experiments

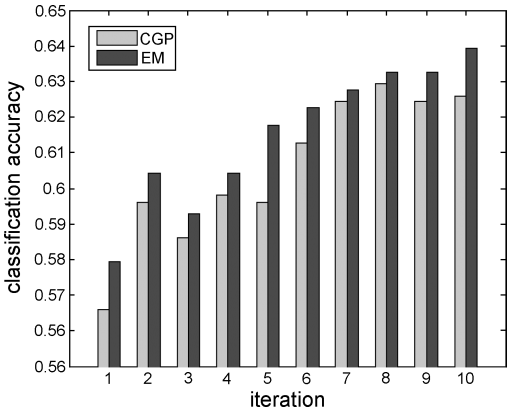


Fig. 13. Classification accuracy for 10 iterations describing EM/CGP cooperation process (*Corel-1200*).

on *Corel-1200*, the total query time for 601 images is ∼6 seconds for RBF-SVM while it is less than 1 second for CFS-EM.

In summary, CFS-EM is much more efficient than RBF-SVM in query phase, which makes it suitable for image retrieval.

5.5 Hill Climbing/CGP Process

As stated in Section 4.3, in CFS-EM, CGP helps EM to jump out of the local maximum. Figure 13 shows a real example on *Corel-1200* dataset (L: 20%, *U*: 80%). In this figure, the *x*-axis is the iteration number, and the y-axis is the classification accuracy on the training data ($L + U$). The *light bars* represent the CGP performance (the best fitness scores) and the *dark bars* represent the EM performance (classification accuracies). The plot shows the performance improvement—the accuracy increases gradually from 56% to 64% in 10 iterations. In iteration 1, CGP gives an initial performance based on the labeled training data L (first light bar); then when bringing in unlabeled training data *U*, EM hill climbs and reaches a local peak (first dark bar), which is ∼1.5% higher than CGP. In the second iteration, CGP helps it to jump out of the local maximum and EM hill climbs to anther local peak. This procedure repeats for 10 iterations. The overall performance is increasing, however the increase for every iteration is not guaranteed even if the previous population is kept as the initialization of the current CGP step, for example, the light bars of iteration 3, 5, and 9 have lower accuracy than their previous light bar. This is because the EM step modifies the classifier and changes the labeling for *U*, which gives a different classifier performance even if the same composite operator vectors are used. Even so, the performance increases in the long-term, which is about 8% after 10 iterations. In this iterative process, CGP serves as both a feature synthesizer and a perturbation factor. It makes CFS-EM jump out of a local maximum to pursue the "possible" global maximum and, thus, makes EM in high dimension possible.

6. CONCLUSIONS

Handling high-dimensional feature vectors in CBIR has been a challenging problem. Unlike other methods to solve this problem, our approach, a CGP/EM hybrid algorithm called *coevolutionary feature synthesized EM* (CFS-EM) algorithm, uses a transductive learning scheme combined with a feature dimensionality reduction technique. In this algorithm, CGP is used to synthesize lower dimensional feature vectors from the original high dimensional feature vectors so that EM can be used to form a Gaussian mixture with limited data. Labeled training data is used to initialize the synthesized feature transformation, and unlabeled training data is used to boost the learned classifier.

In CFS-EM, the Gaussian class distribution in visual feature space is unnecessary. And as a combination of EM and CGP, CFS-EM gives higher classification performance than pure EM [Li et al. 2005], CGP alone and higher performance than other classifier like RBF-SVM, D-EM and TSVM. Moreover, better than a kernel based classifier like RBF-SVM, CFS-EM does not need to specify a kernel function. However, it requires a number of parameters as discussed in Section 4.1.5. But the effect of these parameters is reasonably well understood and the empirical values (Section 5.2) chosen from the experience work fine. As a result, the performance of CFS-EM is in better control. And it gives an explicit transformation from high dimensional visual features to low dimensional synthesized features, which makes retrieval in CBIR more efficient. Our approach overcomes the local maximum problem and provides "near optimal" performance. As a general classification algorithm, our method is not only suitable for CBIR, but also applicable to other pattern recognition and computer vision problems.

As a transductive learning method, our approach obeys the basic assumption that the unlabeled data has a distribution similar to the labeled data. When this assumption is violated, the performance will degrade. To handle the situation when the real number of components is different with the given component number (C in Equation (1)), we can add the component estimation method into the iteration [Dong and Bhanu 2005]. In addition, the predicted labels on the unlabeled patterns have the same weight as the given labels on the labeled patterns. Thus, the predicted labels may dominate in the parameter estimation process of EM, particularly when there are a lot of unlabeled data .[Nigam et al. 2000]. So the use of different weights can possibly make CGP and EM cooperate better and improve the classification accuracy more consistently. An alternative is to introduce the unlabeled data gradually to avoid their dominance on the classifier evaluation. Moreover, introducing relevance feedback and long-term learning will make our system more complete [Dong and Bhanu 2005; Yin et al. 2005]. These will be part of our future work.

REFERENCES

Bargeron, D., Viola, P., and Simard, P. Y. 2005. Boosting-based transductive learning for text detection. In *Proceedings of the International Conference on Document Analysis and Recognition*, 1166–1171.

Bhanu, B. and Lee, S. 1994. *Genetic Learning for Adaptive Image Segmentation*. Kluwer.

Bhanu, B. and Lin, Y. 2003. Genetic algorithm based feature selection for target detection in SAR images. *Image Vision Comput. 21*, 7, 591–608.

Bhanu, B., Lin, Y., and Krawiec, K. 2005. *Evolutionary Synthesis of Pattern Recognition Systems*. Springer-Verlag, Berlin, Germany.

Carson, C., Belongie, S., Greenspan, H., and Malik, J. 2002. Blobworld: Image segmentation using Expectation-Maximization and its application to image querying. *IEEE Trans. Pattern Analys. Mach. Intel. 25*, 8, 1026–1038.

Chang, C. C. and Lin, C. J. 2001. LIBSVM: A library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, Y., Wang, G., and Dong, S. 2003. A progressive transductive inference algorithm based on support vector machine. *J. Softw. 14*, 3, 451–460.

Dempster, A. P., Laird, N. M., and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm with discussion. *J. Royal Statist. Soc. (Series B)* 39, 1, 1–38.

Dong, A. and Bhanu, B. 2003. A new semi-supervised EM algorithm for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, 662–667.

Dong, A. and Bhanu, B. 2005. Active concept learning in image databases. *IEEE Trans. Syst. Manag. Cybernatics, Part B 35*, 3, 450–466.

Dong, A., Bhanu, B., and Lin, Y. 2005. Evolutionary feature synthesis for image databases. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 330–335.

Fang, Y. and Geman, D. 2005. Experiments in mental face retrieval. In *Proceedings of the International Conference on Audio- and Video-Based Biometric Authentication*, 637–646.

Figueriedo, M. A. and Jain, A. K. 2002. Unsupervised learning of finite mixture models. *IEEE Trans. Pattern Analys. Mach. Intel. 24*, 3, 382–396.

FLICKNER, M., SAWHNEY, H. S., NIBLACK, W., ASHLEY, J., HUANG, Q., DOM, B., GORKANI, M., HAFNER, J., LEE, D., PETKOVIC, D., STEELE, D., AND YANKER, P. 1995. Query by image and video content: The QBIC system. *IEEE Trans. Comput. 28*, 9, 23–32.

GUYON, I. AND ELISSEEFF, A. 2003. An introduction to variable and feature selection. *J. Mach. Learn. Resear.* 3, 1157–1182.

HE, X., MA, W. Y., AND ZHANG, H. 2004. Learning an image manifold for retrieval. In *Proceedings of the ACM Multimedia Conference (ACMMM)*, New York, NY, ACM, 17–23.

HE, X., YAN, S., HU, Y., AND ZHANG, H. 2003. Learning a locality preserving subspace for visual recognition. In *Proceedings of the IEEE Conference on Computer Vision*, 385–393.

JOACHIMS, T. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods–Support Vector Learning*. B. Schlkopf, C. Burges, and A. J. Smola, Eds. MIT Press, Cambridge, MA, 169–184.

JOACHIMS, T. 1999. Transductive inference for text classification using support vector machines. In *Proceedings of the International Conference on Machine Learning*.

JOACHIMS, T. 2003. Transductive learning via spectral graph partitioning. In *Proceedings of the International Conference on Machine Learning*, Washington, DC, 290–297.

KOZA, J. R. 1994. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge MA.

LEE, D. D. AND SEUNG, H. S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature 401*, 788–791.

LI, R., BHANU, B., AND DONG, A. 2005. Coevolutionary feature synthesized EM algorithm for image retrieval. In *Proceedings of the ACM Multimedia Conference (ACMMM)*, Singapore, ACM, 696–705.

LI, S. Z., HOU, X., ZHANG, H., AND CHENG, Q. 2001. Learning spatially localized, parts-based representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 207–212.

LIN, Y. AND BHANU, B. 2005. Evolutionary feature synthesis for object recognition. *IEEE Trans. Syst., Manag., Cybernetics, Part C 35*, 2, 156–171.

MCLACHLAN, G. AND PEEL, D. 2000. *Finite Mixture Models*. John Wiley.

MEIR, R. AND RTSCH, G. 2002. An introduction to boosting and leveraging. Lecture Notes in Computer Science. Springer, 118–183.

NAKAZATO, M. AND HUANG, T. S. 2001. 3D Mars: Immersive virtual reality for content-based image retrieval. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, 12–15.

NIGAM, K., MCCALLUM, A. K., THRUN, S., AND MITCHELL, T. 2000. Text classification from labeled and unlabeled documents using EM. *Mach. Learn. 39*, 2–3, 103–134.

PENG, J., BHANU, B., AND QING, S. 1999. Probabilistic feature relevance learning for content-based image retrieval. *Comput. Vision and Image Underst. 75*, 1–2, 150–164.

RUI, Y., HUANG, T. S., ORTEGA, M., AND MEHROTRA, S. 1998. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Trans. Circ. Syst. Video Techn. 8*, 5, 644–655.

SAUNDERS, C., GAMMERMAN, A., AND VOVK, V. 1999. Transduction with confidence and credibility. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers, 722–726.

SMEULDERS, A. W. M., WORRING, M., SANTINI, S., GUPTA, A., AND JAIN, R. 2000. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Analys. Mach. Intel. 22*, 12, 1349–1380.

SU, Z., LI, S., AND ZHANG, H. 2001. Extraction of feature subspaces for content-based retrieval using relevance feedback. In *Proceedings of the ACM Multimedia Conference (ACMMM))*, 98–106.

SWETS, D. L. AND WENG, J. 1999. Hierarchical discriminant analysis for image retrieval. *IEEE Trans. Pattern Analys. Mach. Intel. 21*, 5, 386–401.

TANG, H. L., HANKA, R., AND IP, H. H. S. 2003. Histological image retrieval based on semantic content analysis. *IEEE Trans. Inform. Techn. Biomedicine 7*, 1, 26–36.

VAPNIK, V. 2000. *The Nature of Statistical Learning Theory*. Springer-Verlag. Berlin, Germany.

VIRGINIA, R. D. S. 1993. Learning classification with unlabeled data. In *Advances in Neural Information Processing Systems*. Morgan Kaufmann, 112–119.

WANG, J., LI, J., AND WIEDERHOLD, G. 2001. SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Trans. Pattern Analys. Mach. Intel. 23*, 9, 947–963.

WU, P., MANJUNATH, B. S., AND SHIN, H. D. 2000. Dimensionality reduction for image retrieval. In *Proceedings of the International Conference on Image Processing*, 726–729.

WU, Y. AND HUANG, T. S. 2000. Color tracking by transductive learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 133-138.

WU, Y., TIAN, Q., AND HUANG, T. S. 2000. Discriminant-EM algorithm with application to image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 222–227.

XIANG, S., NIE, F., AND ZHANG, C. 2005. Texture image segmentation: An interactive framework based on adaptive features and transductive learning. In *Proceedings of the Asian Conference on Computer Vision*, 216–225.

XU, W., LIU, X., AND GONG, Y. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of SIGIR*, 267–273.

YIN, P. Y., BHANU, B., CHANG, K. C., AND DONG, A. 2002. Improving retrieval performance by long-term relevance information. In *Proceedings of the IEEE Conference on Compute Vision and Pattern Recognition*, 533–536.

YIN, P. Y., BHANU, B., CHANG, K. C., AND DONG, A. 2005. Integrating relevance feedback techniques for image retrieval using reinforcement learning. *IEEE Trans. Pattern Analys. Mach. Intel. 27*, 10, 1536–1551.

YU, J. AND TIAN, Q. 2006. Learning image manifolds by semantic subspace projection. In *Proceedings of the ACM Multimedia Conference (ACMMM)*, Santa Barbara, CA, 297–306.

ZHOU, X. S., RUI, Y., AND HUANG, T. S. 1999. Water-filling: a novel way for image structural feature extraction. In *Proceedings International Conference on Image Processing*, Kobe, Japan.

ZHU, L. AND ZHANG, A. 2000. Supporting multi-example image queries in image databases. In *Proceedings of the International Conference on Multimedia and Expo*, 397–700.

Received June 2006; revised October 2006, March 2007, June 2007; accepted July 2007