Journal of Machine Learning Research 5 (2004) 913–939 Submitted 7/03; Revised 11/03; Published 8/04

Image Categorization by Learning and Reasoning with Regions

Yixin Chen YIXIN@CS.UNO.EDU
Department of Computer Science
University of New Orleans
New Orleans, LA 70148, USA

James Z. Wang JWANG@IST.PSU.EDU
School of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802, USA

Editor: Donald Geman

Abstract

Designing computer programs to automatically categorize images using low-level features is a challenging research topic in computer vision. In this paper, we present a new learning technique, which extends Multiple-Instance Learning (MIL), and its application to the problem of region-based image categorization. Images are viewed as bags, each of which contains a number of instances corresponding to regions obtained from image segmentation. The standard MIL problem assumes that a bag is labeled positive if at least one of its instances is positive; otherwise, the bag is negative. In the proposed MIL framework, DD-SVM, a bag label is determined by some number of instances satisfying various properties. DD-SVM first learns a collection of *instance prototypes* according to a Diverse Density (DD) function. Each instance prototype represents a class of instances that is more likely to appear in bags with the specific label than in the other bags. A nonlinear mapping is then defined using the instance prototypes and maps every bag to a point in a new feature space, named the *bag feature space*. Finally, standard support vector machines are trained in the bag feature space. We provide experimental results on an image categorization problem and a drug activity prediction problem.

Keywords: image categorization, multiple-instance learning, support vector machines, image classification, image segmentation

1. Introduction

The term image categorization refers to the labeling of images into one of a number of predefined categories. Although this is usually not a very difficult task for humans, it has proved to be an extremely difficult problem for machines (or computer programs). Major sources of difficulties include variable and sometimes uncontrolled imaging conditions, complex and hard-to-describe objects in an image, objects occluding other objects, and the gap between arrays of numbers representing physical images and conceptual information perceived by humans. In this paper, an object in the physical world, which we live in, refers to anything that is visible or tangible and is relatively stable in form. An object in an image is defined as a region, not necessarily connected, which is a projection of an object in the physical world. Designing automatic image categorization algorithms has been an important research field for decades. Potential applications include digital

(a) (b) (c) (d) (e) (f) (g)

Figure 1: Sample images belonging to one of the categories: Mountains and glaciers, Skiing, and Beach.

libraries, Space science, Web searching, geographic information systems, biomedicine, surveillance and sensor systems, commerce, and education.

1.1 Overview of Our Approach

Although color and texture are fundamental aspects for visual perception, human discernment of certain visual contents could be potentially associated with interesting classes of objects or semantic meaning of objects in the image. For one example: if we are asked to decide which images in Figure 1 are images about *Mountains and glaciers*, *Skiing*, and *Beach*, at a single glance, we may come up with the following answers together with supporting arguments:

- Images (a) and (b) are images about mountains and glaciers since we see *mountain* in them;
- Images (c), (d) and (e) are skiing images since there are *snow*, *people*, and perhaps a steep *slope* or *mountain* in them;
- Images (f) and (g) are beach images since we see either *people* playing in *water* or *people* on *sand*;

This seems to be effortless for humans because prior knowledge of similar images and objects may provide powerful assistance for us in recognition. Given a set of labeled images, can a computer program learn such knowledge or semantic concepts from implicit information of objects contained in images? This is the question we attempt to address in this work.

In terms of image representation, our approach is a region-based method. Images are segmented into regions such that each region is roughly homogeneous in color and texture. Each region is characterized by one feature vector describing color, texture, and shape attributes. Consequently, an image is represented by a collection of feature vectors. If segmentation is ideal, regions will correspond to objects. But, in general, semantically accurate image segmentation by a computer program is still an ambitious long-term goal for computer vision researchers (see Shi and Malik, 2000; Wang et al., 2001a; Zhu and Yuille, 1996). Here, semantically accurate image segmentation refers to building a one-to-one mapping between regions generated by an image segmentation algorithm and objects in the image. Nevertheless, we argue that region-based image representation can provide some useful information about objects even though segmentation may not be perfect. Moreover, empirical results in Section 4.3 demonstrate that the proposed method has low sensitivity to inaccurate image segmentation.

From the perspective of learning, our approach is a generalization of supervised learning, in which labels are associated with images instead of individual regions. This is in essence identical to the Multiple-Instance Learning (MIL) setting (Dietterich et al., 1997; Blum and Kalai, 1998; Maron and Lozano-Pérez, 1998; Zhang and Goldman, 2002) where images and regions are respectively called *bags* and *instances*. In this paper, a "bag" refers to an "image", and an "instance" refers to a "region." MIL assumes that every instance possesses an unknown label that is indirectly accessible through labels attached to bags.

1.2 Related Work in Multiple-Instance Learning

Several researchers have applied MIL for image classification and retrieval (Andrews et al., 2003; Maron and Ratan, 1998; Zhang et al., 2002; Yang and Lozano-Pérez, 2000). Key assumptions of their formulation of MIL are that bags and instances share the same set of labels (or categories or classes or topics), and a bag receives a particular label if at least one of the instances in the bag possesses the label. For binary classification, this implies that a bag is "positive" if at least one of its instances is a positive example; otherwise, the bag is "negative." Therefore, learning focuses on finding "actual" positive instances in positive bags. The formulations of MIL in image classification and retrieval fall into two categories: the Diverse Density approach (Maron and Ratan, 1998; Zhang et al., 2002) and the Support Vector Machine (SVM) approach (Andrews et al., 2003).

- In the Diverse Density approach, an objective function, called the Diverse Density (DD) function (Maron and Lozano-Pérez, 1998), is defined over the instance feature space, in which instances can be viewed as points. The DD function measures a co-occurrence of similar instances from different bags with the same label. A feature point with large Diverse Density indicates that it is close to at least one instance from every positive bag and far away from every negative instance. The DD approach searches the instance feature space for points with high Diverse Density. Once a point with the maximum DD is found, a new bag is classified according to the distances between instances in the bag and the maximum DD point: if the smallest distance is less than certain fixed threshold, the bag is classified as positive; otherwise, the bag is classified as negative. The major difference between Maron's method and Zhang's method lies in the way to search a maximum DD point. Zhang's method is relatively insensitive to the dimension of instance feature space and scales up well to the average bag size, i.e., the average number of instances in a bag (Zhang and Goldman, 2002). Empirical studies demonstrate that DD-based MIL can learn certain simple concepts of natural scenes, such as *waterfall*, *sunsets*, and *mountains*, using features of subimages or regions (Maron and Ratan, 1998; Zhang et al., 2002).

- Andrews et al. (2003) use SVMs (Burges, 1998) to solve the MIL problem. In particular, MIL is formulated as a mixed integer quadratic program. In their formulation, integer variables are selector variables that select which instance in a positive bag is the positive instance. Their algorithm, which is called MI-SVM, has an outer loop and an inner loop. The outer loop sets the values of these selector variables. The inner loop then trains a standard SVM in which the selected positive instances replace the positive bags. The outer loop stops if none of the selector variables changes value in two consecutive iterations. Andrews et al. (2003) show that MI-SVM outperforms the DD approach described in Zhang and Goldman (2002) on a set of images belonging to three different categories ("elephant", "fox", and "tiger"). The difference between MI-SVM and DD approach can also be viewed from the shape of the corresponding classifier's decision boundary in the instance feature space. The decision boundary of a DD classifier is an ellipsoidal sphere because classification is based exclusively on the distance to the maximum DD point.[1] For MI-SVM, depending on the kernel used, the decision boundary can be a hyperplane in the instance feature space or a hyperplane in the kernel induced feature space, which may correspond to very complex boundaries in the instance feature space.

1.3 A New Formulation of Multiple-Instance Learning

In the above MIL formulations, a bag is essentially summarized by one of its instances, i.e., an instance with the maximal label (considering binary classification with 1 and -1 representing the positive and negative classes, respectively). However, these formulations have a drawback for image categorization tasks: in general, a concept about images may not be captured by a single region (instance) even if image segmentation and object recognition are assumed to be ideal (inaccurate segmentation and recognition will only worsen the situation). For one simple example, let's consider categorizing *Mountains and glaciers* versus *Skiing* images in Figure 1. To classify a scene as involving skiing, it is helpful to identify *snow*, *people*, and perhaps *mountain*. If an image is viewed as a bag of regions, then the standard MIL formulation cannot realize this, because a bag is labeled positive if any one region in the bag is positive. In addition, a class might also be disjunctive. As shown by Figure 1 (g) and (h), a *Beach* scene might involve either *people* playing in *water* or *people* on *sand*. Thus we argue that the correct categorization of an image depends on identifying multiple aspects of the image. This motivates our extension of MIL where a bag must contain a number of instances satisfying various properties (e.g., *people*, *snow*, etc.).

In our approach, MIL is formulated as a maximum margin problem in a new feature space defined by the DD function. The new approach, named DD-SVM, proceeds in two steps. First, in the instance feature space, a collection of feature vectors, each of which is called an *instance prototype*, is determined according to DD. Each instance prototype is chosen to be a local maximizer of the DD function. Since DD measures the co-occurrence of similar instances from different bags with the same label, loosely speaking, an instance prototype represents a class of instances (or regions) that is more likely to appear in bags (or images) with the specific label than in the other bags (or images). Second, a nonlinear mapping is defined using the learned instance prototypes and maps every bag to a point in a new feature space, which is named the *bag feature space*. In the bag feature space, the original MIL problem becomes an ordinary supervised learning problem. Standard SVMs are then trained in the bag feature space.

DD-SVM is similar to MI-SVM in the sense that both approaches apply SVM to solve the MIL problem. However, in DD-SVM, several features are defined for each bag. Each bag feature could be defined by a separate instance within the bag (i.e., the instance that is most similar to an instance prototype). Hence, the bag features summarize the bag along several dimensions defined by instance prototypes. This is in stark contrast to MI-SVM, in which one instance is selected to represent the whole positive bag.

1.4 Related Work in Image Categorization

In the areas of image processing, computer vision, and pattern recognition, there has been abundance of prior work on detecting, recognizing, and classifying a relatively small set of objects or concepts

1. The maximum DD algorithms described in (Maron and Lozano-Pérez, 1998; Zhang and Goldman, 2002) produce a point in the instance feature space together with scaling factors for each feature dimension. Therefore, the decision boundary is an ellipsoidal sphere instead of a sphere.

in specific domains of application (Forsyth and Ponce, 2002; Marr, 1983; Strat, 1992). We only review work most relevant to what we propose, which by no means represents the comprehensive list in the cited area.

As one of the simplest representations of digital images, histograms have been widely used for various image categorization problems. Szummer and Picard (1998) use k-nearest neighbor classifier on color histograms to discriminate between *indoor* and *outdoor* images. In the work of Vailaya et al. (2001), Bayesian classifiers using color histograms and edge directions histograms are implemented to organize *sunset/forest/mountain* images and *city/landscape* images, respectively. Chapelle et al. (1999) apply SVMs, which are built on color histogram features, to classify images containing a generic set of objects. Although histograms can usually be computed with little cost and are effective for certain classification tasks, an important drawback of a global histogram representation is that information about spatial configuration is ignored. Many approaches have been proposed to tackle the drawback. In the method of Huang et al. (1998), a classification tree is constructed using color correlograms. Color correlogram captures the spatial correlation of colors in an image. Gdalyahu and Weinshall (1999) apply local curve matching for shape silhouette classifications, in which objects in images are represented by their outlines.

A number of subimage-based methods have been proposed to exploit local and spatial properties by dividing an image into rectangular blocks. In the method introduced by Gorkani and Picard (1994), an image is first divided into 16 non-overlapping equal-sized blocks. Dominant orientations are computed for each block. The image is then classified as *city* or *suburb* scenes as determined by the majority orientations of blocks. Wang et al. (2001b) develop a *graph/photograph* classification algorithm.[2] The classifier partitions an image into blocks and classifies every block into one of two categories based on wavelet coefficients in high frequency bands. If the percentage of blocks classified as photograph is higher than a threshold, the image is marked as a photograph; otherwise, the image is marked as a graph. Yu and Wolf (1995) present a one-dimensional Hidden Markov Model (HMM) for *indoor/outdoor* scene classification. The model is trained on vector quantized color histograms of image blocks. In the recent ALIP system (Li and Wang, 2003), a concept corresponding to a particular category of images is captured by a two-dimensional multiresolution HMM trained on color and texture features of image blocks. Murphy et al. (2004) propose four graphical models that relate features of image blocks to objects and perform joint scene and object recognition.

Although a rigid partition of an image into rectangular blocks preserves certain spatial information, it often breaks an object into several blocks or puts different objects into a single block. Thus visual information about objects, which could be beneficial to image categorization, may be destroyed by a rigid partition. The ALIP system (Li and Wang, 2003) uses a small block size (4×4) for feature extraction to avoid this problem. Image segmentation is one way to extract object information. It decomposes an image into a collection of regions, which correspond to objects if decomposition is ideal. Segmentation-based algorithms can take into consideration the shape information, which is in general not available without segmentation.

Image segmentation has been successfully used in content-based image and video analysis (e.g., Carson et al., 2002; Chen and Wang, 2002; Ma and Manjunath, 1997; Modestino and Zhang, 1992; Smith and Li, 1999; Vasconcelos and Lippman, 1998; Wang et al., 2001b). Modestino and Zhang (1992) apply a Markov random field model to capture spatial relationships between regions. Im-

2. As defined by Wang et al. (2001b), a graph image is an image containing mainly text, graph, and overlays; a photograph is a continuous-tone image.

age interpretation is then given by a maximum *a posteriori* rule. SIMPLIcity system (Wang et al., 2001b) classifies images into *textured* or *nontextured* classes based upon how evenly a region scatters in an image. Mathematically, this is described by the goodness of match, which is measured by the χ^2 statistics, between the distribution of the region and a uniform distribution. Smith and Li (1999) propose a method for classifying images by spatial orderings of regions. Their system decomposes an image into regions with the attribute of interest of each region represented by a symbol that corresponds to an entry in a finite pattern library. The region string is converted to composite region template descriptor matrix that enables classification using spatial information. Vasconcelos and Lippman (1998) model image retrieval as a classification problem based on the principle of Bayesian inference. The information of the regions identified as human skin is used in the inference. Very interesting results have been achieved in associating words to images based on regions (Barnard and Forsyth, 2001) or relating words to image regions (Barnard et al., 2003). In their method, an image is modeled as a sequence of regions and a sequence of words generated by a hierarchical statistic model. The method demonstrates the potential for searching images. But as noted by Barnard and Forsyth (2001), the method relies on semantically meaningful segmentation, which, as mentioned earlier, is still an open problem in computer vision.

1.5 Outline of the Paper

The remainder of the paper is organized as follows. Section 2 describes image segmentation and feature representation. Section 3 presents DD-SVM, an extension of MIL. Section 4 describes the extensive experiments we have performed and provides the results. Finally, we conclude in Section 5, together with a discussion of future work.

2. Image Segmentation and Representation

In this section we describe a simple image segmentation procedure based on color and spatial variation features using a *k*-means algorithm (Hartigan and Wong, 1979). For general-purpose images such as the images in a photo library or images on the World Wide Web, precise object segmentation is nearly as difficult as natural language semantics understanding. However, semantically precise segmentation is not crucial to our system. As we will demonstrate in Section 4, our image categorization method has low sensitivity to inaccurate segmentation. Image segmentation is a well-studied topic (e.g., Shi and Malik, 2000; Wang et al., 2001a; Zhu and Yuille, 1996). The focus of this paper is not to achieve superior segmentation results but good categorization performance. The major advantage of the proposed image segmentation is its low computational cost.

To segment an image, the system first partitions the image into non-overlapping blocks of size 4×4 pixels. A feature vector is then extracted for each block. The block size is chosen to compromise between texture effectiveness and computation time. Smaller block size may preserve more texture details but increase the computation time as well. Conversely, increasing the block size can reduce the computation time but lose texture information and increase the segmentation coarseness. Each feature vector consists of six features. Three of them are the average color components in a block. We use the well-known LUV color space, where L encodes luminance and U and V encode color information (chrominance). The other three represent square root of energy in the high-frequency bands of the wavelet transforms (Daubechies, 1992), that is, the square root of the second order moment of wavelet coefficients in high-frequency bands.

Figure 2: Segmentation results by the k-means clustering algorithm. First row: original images. Second row: regions in their representative colors.

To obtain these moments, a Daubechies-4 wavelet transform is applied to the L component of the image. After a one-level wavelet transform, a 4×4 block is decomposed into four frequency bands: the LL, LH, HL, and HH bands. Each band contains 2×2 coefficients. Without loss of generality, we suppose the coefficients in the HL band are $\{c_{k,l}, c_{k,l+1}, c_{k+1,l}, c_{k+1,l+1}\}$. One feature is

$$f = \left(\frac{1}{4} \sum_{i=0}^{1} \sum_{j=1}^{1} c_{k+i,l+j}^2 \right)^{\frac{1}{2}} .$$

The other two features are computed similarly to the LH and HH bands. Unser (1995) shows that moments of wavelet coefficients in various frequency bands are effective for representing texture. For example, the HL band shows activities in the horizontal direction. An image with vertical strips thus has high energy in the HL band and low energy in the LH band.

The k-means algorithm is used to cluster the feature vectors into several classes with every class corresponding to one "region" in the segmented image. No information about the spatial layout of the image is used in defining the regions, so they are not necessarily spatially contiguous. The algorithm does not specify the number of clusters, N, to choose. We adaptively select N by gradually increasing N until a stopping criterion is met. The number of clusters in an image changes in accordance with the adjustment of the stopping criteria. A detailed description of the stopping criteria can be found in Wang et al. (2001b). Examples of segmentation results are shown in Figure 2. Segmented regions are shown in their representative colors. It takes less than one second on average to segment a 384×256 image on a Pentium III 700MHz PC running the Linux operating system. Since it is almost impossible to find a stopping criterion that is best suited for a large collection of images, images sometimes may be under-segmented or over-segmented. However, our categorization method has low sensitivity to inaccurate segmentation.

After segmentation, the mean of the set of feature vectors corresponding to each region $\mathbf{R}_j$ (a subset of $\mathbb{Z}^2$) is computed and denoted as $\hat{\mathbf{f}}_j$. Three extra features are also calculated for each region to describe shape properties. They are normalized inertia (Gersho, 1979) of order 1, 2, and 3. For a region $\mathbf{R}_j$ in the image plane, the normalized inertia of order γ is given as

$$I(\mathbf{R}_j,\gamma) = \frac{\sum_{\mathbf{r}\in\mathbf{R_j}} \|\mathbf{r}-\hat{\mathbf{r}}\|^{\gamma}}{V_j^{1+\frac{\gamma}{2}}} ,$$

where $\hat{\mathbf{r}}$ is the centroid of $\mathbf{R}_j$, V_j is the number of pixels in region $\mathbf{R}_j$. The normalized inertia is invariant to scaling and rotation. The minimum normalized inertia on a 2-dimensional plane is achieved by circles. Denote the γ-th order normalized inertia of circles as I_γ. We define shape features of region $\mathbf{R}_j$ as

$$\mathbf{s}_j = \left[\frac{I(\mathbf{R}_j,1)}{I_1}, \frac{I(\mathbf{R}_j,2)}{I_2}, \frac{I(\mathbf{R}_j,3)}{I_3}\right]^T .$$

Finally, an image $\mathbf{B}_i$, which is segmented into N_i regions $\{\mathbf{R}_j : j = 1,\cdots,N_i\}$, is represented by a collection of feature vectors $\{\mathbf{x}_{ij} : j = 1,\cdots,N_i\}$. Each $\mathbf{x}_{ij}$ is a 9-dimensional feature vector, corresponding to region $\mathbf{R}_j$, defined as

$$\mathbf{x}_{ij} = \left[\hat{\mathbf{f}}_j^T, \mathbf{s}_j^T\right]^T .$$

3. An Extension of Multiple-Instance Learning

In this section, we first introduce DD-SVM, a maximum margin formulation of MIL in bag feature space. We then describe one way to construct a bag feature space using Diverse Density. Finally, we compare DD-SVM with another SVM-based MIL formulation, MI-SVM, proposed by Andrews et al. (2003).

3.1 Maximum Margin Formulation of MIL in a Bag Feature Space

We start with some notations in MIL. Let $\mathcal{D}$ be the labeled data set, which consists of l bag/label pairs, i.e., $\mathcal{D} = \{(\mathbf{B}_1,y_1),\cdots,(\mathbf{B}_l,y_l)\}$. Each bag $\mathbf{B}_i \subset \mathbb{R}^m$ is a collection of instances with $\mathbf{x}_{ij} \in \mathbb{R}^m$ denoting the j-th instance in the bag. Different bags may have different number of instances. Labels y_i take binary values 1 or -1. A bag is called a positive bag if its label is 1; otherwise, it is called a negative bag. Note that a label is attached to each bag and not to every instance. In the context of images, a bag is a collection of region feature vectors; an instance is a region feature vector; positive (negative) label represents that an image belongs (does not belong) to a particular category.

The basic idea of the new MIL framework is to map every bag to a point in a new feature space, named the bag feature space, and to train SVMs in the bag feature space. For an introduction to SVMs, we refer interested readers to tutorials and books on this topic (Burges, 1998; Cristianini and Shawe-Taylor, 2000). The maximum margin formulation of MIL in a bag feature space is given as the following quadratic optimization problem:

$$DD-SVM \qquad \alpha^* = \arg\max_{\alpha_i} \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K(\phi(\mathbf{B}_i),\phi(\mathbf{B}_j)) \tag{1}$$

$$subject\ to \quad \sum_{i=1}^{l} y_i\alpha_i = 0$$

$$C \geq \alpha_i \geq 0,\ i = 1,\cdots,l.$$

The bag feature space is defined by $\phi : \mathcal{B} \rightarrow \mathbb{R}^n$ where $\mathcal{B}$ is a subset of $\mathcal{P}(\mathbb{R}^m)$ (the power set of $\mathbb{R}^m$). In practice, we can assume that the elements of $\mathcal{B}$ are finite sets since the number of instances in a bag is finite. $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is a kernel function. The parameter C controls the trade-off between accuracy and regularization. The bag classifier is then defined by α^* as

$$\text{label}(\mathbf{B}) = \text{sign}\left(\sum_{i=1}^{l} y_i \alpha_i^* K(\phi(\mathbf{B}_i), \phi(\mathbf{B})) + b^*\right) \tag{2}$$

where b^* is chosen so that

$$y_j \left(\sum_{i=1}^{l} y_i \alpha_i^* K(\phi(\mathbf{B}_i), \phi(\mathbf{B}_j)) + b^*\right) = 1$$

for any j with $C > \alpha_j^* > 0$. The optimization problem (1) assumes that the bag feature space (i.e., ϕ) is given. Next, we introduce a way of constructing ϕ from a set of labeled bags.

3.2 Constructing a Bag Feature Space

Given a set of labeled bags, finding what is in common among the positive bags and does not appear in the negative bags may provide inductive clues for classifier design. In our approach, such clues are captured by instance prototypes computed from the DD function. A bag feature space is then constructed using the instance prototypes, each of which defines one dimension of the bag feature space.

3.2.1 DIVERSE DENSITY

In the ideal scenario, the intersection of the positive bags minus the union of the negative bags gives the instances that appear in all the positive bags but none of the negative bags. However, in practice strict set operations of intersection, union, and difference may not be useful because most real world problems involve noisy information. Features of instances might be corrupted by noise. Some bags might be mistakenly labeled. Strict intersection of positive bags might generate the empty set. Diverse Density implements soft versions of the intersection, union, and difference operations by thinking of the instances and bags as generated by some probability distribution. It is a function defined over the instance feature space. The DD value at a point in the feature space is indicative of the probability that the point agrees with the underlying distribution of positive and negative bags.

Next, we introduce one definition of DD from Maron and Lozano-Pérez (1998). Interested readers are referred to Maron and Lozano-Pérez (1998) for detailed derivations based on a probabilistic framework. Given a labeled data set $\mathcal{D}$, the DD function is defined as

$$DD_{\mathcal{D}}(\mathbf{x}, \mathbf{w}) = \prod_{i=1}^{l} \left[\frac{1+y_i}{2} - y_i \prod_{j=1}^{N_i} \left(1 - e^{-\|\mathbf{x}_{ij} - \mathbf{x}\|_{\mathbf{w}}^2}\right)\right]. \tag{3}$$

Here, $\mathbf{x}$ is a point in the instance feature space; $\mathbf{w}$ is a weight vector defining which features are considered important and which are considered unimportant; N_i is the number of instances in the i-th bag; and $\|\cdot\|_{\mathbf{w}}$ denotes a weighted norm defined by

$$\|\mathbf{x}\|_{\mathbf{w}} = \left[\mathbf{x}^T \text{Diag}(\mathbf{w})^2 \mathbf{x}\right]^{\frac{1}{2}} \tag{4}$$

where $\mathrm{Diag}(\mathbf{w})$ is a diagonal matrix whose (i,i)-th entry is the i-th component of $\mathbf{w}$.

It is not difficult to observe that values of DD are always between 0 and 1. For fixed weights $\mathbf{w}$, if a point $\mathbf{x}$ is close to an instance from a positive bag $\mathbf{B}_i$, then

$$\frac{1+y_i}{2} - y_i \prod_{j=1}^{N_i} \left(1 - e^{-\|\mathbf{x}_{ij}-\mathbf{x}\|_{\mathbf{w}}^2}\right) \tag{5}$$

will be close to 1; if $\mathbf{x}$ is close to an instance from a negative bag $\mathbf{B}_i$, then (5) will be close to 0. The above definition indicates that $DD(\mathbf{x},\mathbf{w})$ will be close to 1 if $\mathbf{x}$ is close to instances from different positive bags and, at the same time, far away from instances in all negative bags. Thus it measures a co-occurrence of instances from different (diverse) positive bags.

3.2.2 Learning Instance Prototypes

The DD function defined in (3) is a continuous and highly nonlinear function with multiple peaks and valleys (or local maximums and minimums). A larger value of DD at a point indicates a higher probability that the point fits better with the instances from positive bags than with those from negative bags. This motivates us to choose local maximizers of DD as instance prototypes. Loosely speaking, an instance prototype represents a class of instances that is more likely to appear in positive bags than in negative bags. Note that, the MIL formulation in Maron and Lozano-Pérez (1998) computes the global maximizer of DD, which corresponds to one instance prototype in our notation.

Learning instance prototypes then becomes an optimization problem: finding local maximizers of the DD function in a high-dimensional space. For our application the dimension of the optimization problem is 18 because the dimension of the region features is 9 and the dimension of weights is also 9. Since the DD functions are smooth, we apply gradient based methods to find local maximizers. Now the question is: how do we find all the local maximizers? In general, we do not know the number of local maximizers a DD function has. However, according to the definition of DD, a local maximizer is close to instances from positive bags (Maron and Lozano-Pérez, 1998). Thus starting a gradient based optimization from one of those instances will likely lead to a local maximum. Therefore, a simple heuristic is applied to search for multiple maximizers: we start an optimization at every instance in every positive bag with uniform weights, and record all the resulting distinct maximizers (feature vector and corresponding weights).

Instance prototypes are selected from those maximizers with two additional constraints: (a) they need to be distinct from each other; and (b) they need to have large DD values. The first constraint addresses the precision issue of numerical optimization. Due to numerical precision, different starting points may lead to different versions of the same maximizer. Hence we need to remove some of the maximizers that are essentially repetitions of each other. The second constraint limits instance prototypes to those that are most informative in terms of co-occurrence in different positive bags. In our algorithm, this is achieved by picking maximizers with DD values greater than certain threshold.

Following the above descriptions, one can find instance prototypes representing classes of instances that are more likely to appear in positive bags than in negative bags. One could argue that instance prototypes with the exactly reversed property (more likely to appear in negative bags than in positive bags) may be of equal importance. Such instance prototypes can be computed in exactly the same fashion after negating the labels of positive and negative bags. Our empirical study shows that including such instance prototypes (for negative bags) improves classification accuracy by an average amount of 2.2% for the 10-class image categorization experiment described in Section 4.2.

3.2.3 AN ALGORITHMIC VIEW

Next, we summarize the above discussion in pseudo code. The input is a set of labeled bags $\mathcal{D}$. The following pseudo code learns a collection of instance prototypes each of which is represented as a pair of vectors $(\mathbf{x}_i^*, \mathbf{w}_i^*)$. The optimization problem involved is solved by Quasi-Newton search `dfpmin` in Press et al. (1992).

Algorithm 3.1 Learning Instance Prototypes

MainLearnIPs($\mathcal{D}$)

1 $\mathbf{I}_p$ = **LearnIPs**($\mathcal{D}$) [*Learn Instance Prototypes for positive bags*]
2 negate labels of all bags in $\mathcal{D}$
3 $\mathbf{I}_n$ = **LearnIPs**($\mathcal{D}$) [*Learn Instance Prototypes for negative bags*]
4 OUTPUT (the set union of $\mathbf{I}_p$ and $\mathbf{I}_n$)

LearnIPs($\mathcal{D}$)

1 set $\mathbf{P}$ be the set of instances from all positive bags in $\mathcal{D}$
2 initialize $\mathbf{M}$ to be the empty set
3 FOR (every instance in $\mathbf{P}$ as starting point for $\mathbf{x}$)
4   set the starting point for $\mathbf{w}$ to be all $1'$s
5   find a maximizer $(\mathbf{p}, \mathbf{q})$ of the $\log(DD)$ function by quasi-Newton search
6   add $(\mathbf{p}, \mathbf{q})$ to $\mathbf{M}$
7 END
8 set $i = 1$, $T = \frac{\max_{(\mathbf{p},\mathbf{q})\in\mathbf{M}} log(DD_{\mathcal{D}}(\mathbf{p},\mathbf{q})) + \min_{(\mathbf{p},\mathbf{q})\in\mathbf{M}} log(DD_{\mathcal{D}}(\mathbf{p},\mathbf{q}))}{2}$
9 REPEAT
10   set $(\mathbf{x}_i^*, \mathbf{w}_i^*) = \arg\max_{(\mathbf{p},\mathbf{q})\in\mathbf{M}} \log(DD_{\mathcal{D}}(\mathbf{p}, \mathbf{q}))$
11   remove from $\mathbf{M}$ all elements $(\mathbf{p}, \mathbf{q})$ satisfying
   $\|\mathbf{p} \otimes \text{abs}(\mathbf{q}) - \mathbf{x}_i^* \otimes \text{abs}(\mathbf{w}_i^*)\| < \beta \|\mathbf{x}_i^* \otimes \text{abs}(\mathbf{w}_i^*)\|$ OR $\log(DD_{\mathcal{D}}(\mathbf{p}, \mathbf{q})) < T$
12   set $i = i + 1$
13 WHILE ($\mathbf{M}$ is not empty)
14 OUTPUT ($\{(\mathbf{x}_1^*, \mathbf{w}_1^*), \cdots, (\mathbf{x}_{i-1}^*, \mathbf{w}_{i-1}^*)\}$)

In the above pseudo code for **LearnIPs**, lines 1–7 find a collection of local maximizers for the DD function by starting optimization at every instance in every positive bag with uniform weights. For better numerical stability, the optimization is performed on the $\log(DD)$ function, instead of the DD function itself. In line 5, we implement the EM-DD algorithm (Zhang and Goldman, 2002), which scales up well to large bag sizes in running time. Lines 8–13 describe an iterative process to pick a collection of "distinct" local maximizers as instance prototypes. In each iteration, an element of $\mathbf{M}$, which is a local maximizer, with the maximal $\log(DD)$ value (or, equivalently, the DD value) is selected as an instance prototype (line 10). Then elements of $\mathbf{M}$ that are close to the selected instance prototype or that have DD values lower than a threshold are removed from $\mathbf{M}$ (line 11). A new iteration starts if $\mathbf{M}$ is not empty. The $\text{abs}(\mathbf{w})$ in line 11 computes component-wise absolute values of $\mathbf{w}$. This is because the signs in $\mathbf{w}$ have no effect on the definition (4) of weighted norm. The $\otimes$ in line 11 denotes component-wise product.

The number of instance prototypes selected from $\mathbf{M}$ is determined by two parameters β and T. In our implementation, β is set to be 0.05, and T is the average of the maximal and minimal $\log(DD)$

values for all local maximizers found (line 8). These two parameters may need to be adjusted for other applications. However, empirical study shows that the performance of the classifier is not sensitive to β and T. Experimental analysis of the conditions under which the algorithm will find good instance prototypes is given in Section 4.5.

3.2.4 Computing Bag Features

Let $\{(\mathbf{x}_k^*, \mathbf{w}_k^*) : k = 1, \cdots, n\}$ be the collection of instance prototypes given by Algorithm 3.1. We define bag features, $\phi(\mathbf{B}_i)$, for a bag $\mathbf{B}_i = \{\mathbf{x}_{ij} : j = 1, \cdots, N_i\}$, as

$$\phi(\mathbf{B}_i) = \begin{bmatrix} \min_{j=1,\cdots,N_i} \|\mathbf{x}_{ij} - \mathbf{x}_1^*\|_{\mathbf{w}_1^*} \\ \min_{j=1,\cdots,N_i} \|\mathbf{x}_{ij} - \mathbf{x}_2^*\|_{\mathbf{w}_2^*} \\ \vdots \\ \min_{j=1,\cdots,N_i} \|\mathbf{x}_{ij} - \mathbf{x}_n^*\|_{\mathbf{w}_n^*} \end{bmatrix} . \tag{6}$$

In the definition (6), each bag feature is defined by one instance prototype and one instance from the bag, i.e., the instance that is "closest" to the instance prototype. A bag feature gives the smallest distance (or highest similarity score) between any instance in the bag and the corresponding instance prototype. Hence, it can also be viewed as a measure of the degree that an instance prototype shows up in the bag.

3.3 Comparing DD-SVM with MI-SVM

The following pseudo code summarizes the learning process of DD-SVM. The input is $\mathcal{D}$, a collection of bags with binary labels. The output is an SVM classifier defined by (2).

Algorithm 3.2 Learning DD-SVM

DD-SVM($\mathcal{D}$)

```
let S be the empty set
IP = MainLearnIPs(D)
FOR (every bag B in D)
   define bag features φ(B) according to (6)
   add (φ(B),y) to S where y is the label of B
END
train a standard SVM using S
OUTPUT (the SVM)
```

MI-SVM, proposed by Andrews et al. (2003), is also an SVM-based MIL method. In Section 4, we experimentally compare DD-SVM against MI-SVM. An algorithmic description of MI-SVM is given below. The input is a collection of labeled bags $\mathcal{D}$. The output is a classifier of the form

$$\text{label}(\mathbf{B}_i) = \text{sign}\left(max_{j=1,\cdots,N_i} f(\mathbf{x}_{ij})\right) \tag{7}$$

where $\mathbf{x}_{ij}$, $j = 1, \cdots, N_i$, are instances of $\mathbf{B}_i$, f is a function given by SVM learning.

Algorithm 3.3 Learning MI-SVM

MI-SVM($\mathcal{D}$)
1 let $\mathbf{P}$ be the empty set
2 FOR (every positive bag $\mathbf{B}$ in $\mathcal{D}$)
3   set $\mathbf{x}^*$ be the average of instances in $\mathbf{B}$
4   add $(\mathbf{x}^*, 1)$ to $\mathbf{P}$
5 END
6 let $\mathbf{N}$ be the empty set
7 FOR (every negative bag $\mathbf{B}$ in $\mathcal{D}$)
8   FOR (every instance $\mathbf{x}$ in $\mathbf{B}$)
9   add $(\mathbf{x}, -1)$ to $\mathbf{N}$
10   END
11 END
12 REPEAT
13   set $\mathbf{P}' = \mathbf{P}$
14   set $\mathbf{S} = \mathbf{P}' \cup \mathbf{N}$
15   train a standard SVM, $\mathrm{label}(\mathbf{x}) = \mathrm{sign}(f(\mathbf{x}))$, using $\mathbf{S}$
16   let $\mathbf{P}$ be the empty set
17   FOR (every positive bag $\mathbf{B}$ in $\mathcal{D}$)
18   set $\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathbf{B}} f(\mathbf{x})$
19   add $(\mathbf{x}^*, 1)$ to $\mathbf{P}$
20   END
21 WHILE $(\mathbf{P} \neq \mathbf{P}')$
22 OUTPUT (the classifier defined by (7))

In the above pseudo code for MI-SVM, the key steps are the loop given by lines 12–21. During each iteration, a standard SVM classifier, $\mathrm{label}(\mathbf{x}) = \mathrm{sign}(f(\mathbf{x}))$, is trained in the instance space. The training set is the union of negative instances and positive instances. Negative instances are those from every negative bag. Each positive instance represents a positive bag. It is chosen to be the instance, in a positive bag, with the maximum f value from the previous iteration. In the first iteration, each positive instance is initialized to be the average of the feature vectors in the bag. The loop terminates if the set of positive instances selected for the next iteration is identical to that of the current iteration.

The crucial difference between DD-SVM and MI-SVM lies in the underlying assumption. MI-SVM method, as well as other standard MIL methods (such as the DD approach proposed by Maron and Lozano-Pérez, 1998), assumes that if a bag is labeled negative then all instances in that bag is negative, and if a bag is labeled positive, then as least one of the instances in that bag is a positive instance. In MI-SVM, one instance is selected to represent the whole positive bag. An SVM is trained in the instance feature space using all negative instances and the selected positive instances. Our DD-SVM method assumes that a positive bag must contain some number of instances satisfying various properties, which are captured by bag features. Each bag feature is defined by an instance in the bag and an instance prototype derived from the DD function. Hence, the bag features summarize the bag along several dimensions. An SVM is then trained in the bag feature space.

4. Experiments

We present systematic evaluations of DD-SVM based on a collection of images from the COREL and the MUSK data sets. The data sets and the source code of DD-SVM can be downloaded at `http://www.cs.uno.edu/~yixin/ddsvm.html`. Section 4.1 describes the experimental setup for image categorization, including the image data set, the implementation details, and the selection of parameters. Section 4.2 compares DD-SVM with MI-SVM and color histogram-based SVM using COREL data. The effect of inaccurate image segmentation on classification accuracies is demonstrated in Section 4.3. Section 4.4 illustrates the performance variations when the number of image categories increases. Analysis of the effects of training sample size and diversity of images is given in Section 4.5. Results on the MUSK data sets are presented in Section 4.6. Computational issues are discussed in Section 4.7.

4.1 Experimental Setup for Image Categorization

The image data set employed in our empirical study consists of $2,000$ images taken from 20 CD-ROMs published by COREL Corporation. Each COREL CD-ROM of 100 images represents one distinct topic of interest. Therefore, the data set has 20 thematically diverse image categories, each containing 100 images. All the images are in JPEG format with size 384×256 or 256×384. We assigned a keyword (or keywords) to describe each image category. The category names and some randomly selected sample images from each category are shown in Figure 3.

Images within each category are randomly divided into a training set and a test set each with 50 images. We repeat each experiment for 5 random splits, and report the average of the results obtained over 5 different test sets together with the 95% confidence interval. The SVM^{Light} (Joachims, 1999) software is used to train the SVMs. The classification problem here is clearly a multi-class problem. We use the one-against-the-rest approach: (a) for each category, an SVM is trained to separate that category from all the other categories; (b) the final predicted class label is decided by the winner of all SVMs, i.e., one with the maximum value inside the $\text{sign}(\cdot)$ function in (2).

Two other image classification methods are implemented for comparison. One is a histogram-based SVM classification approach proposed by Chapelle et al. (1999). We denote it by Hist-SVM. Each image is represented by a color histogram in the LUV color space. The dimension of each histogram is 125. The other is MI-SVM (Andrews et al., 2003). MI-SVM uses the same set of region features as our approach (it is implemented according to the pseudo code in Algorithm 3.3). The learning problems in Hist-SVM and MI-SVM are solved by SVM^{Light}. The Gaussian kernel, $K(\mathbf{x},\mathbf{z}) = e^{-s\|\mathbf{x}-\mathbf{z}\|^2}$, is used in all three methods.

Several parameters need to be specified for SVM^{Light}.[3] The most significant ones are s and C (the constant in (1) controlling the trade-off between training error and regularization). We apply the following strategy to select these two parameters: We allow each one of the two parameters be respectively chosen from two sets each containing 10 predetermined numbers. For every pair of values of the two parameters (there are 100 pairs in total), a twofold cross-validation error on the training set is recorded. The pair that gives the minimum twofold cross-validation error is selected to be the "optimal" parameters. Note that the above procedure is applied only once for each method. Once the parameters are determined, they are used in all subsequent image categorization experiments.

3. SVM^{Light} software and detailed descriptions of all its parameters are available at `http://svmlight.joachims.org`.

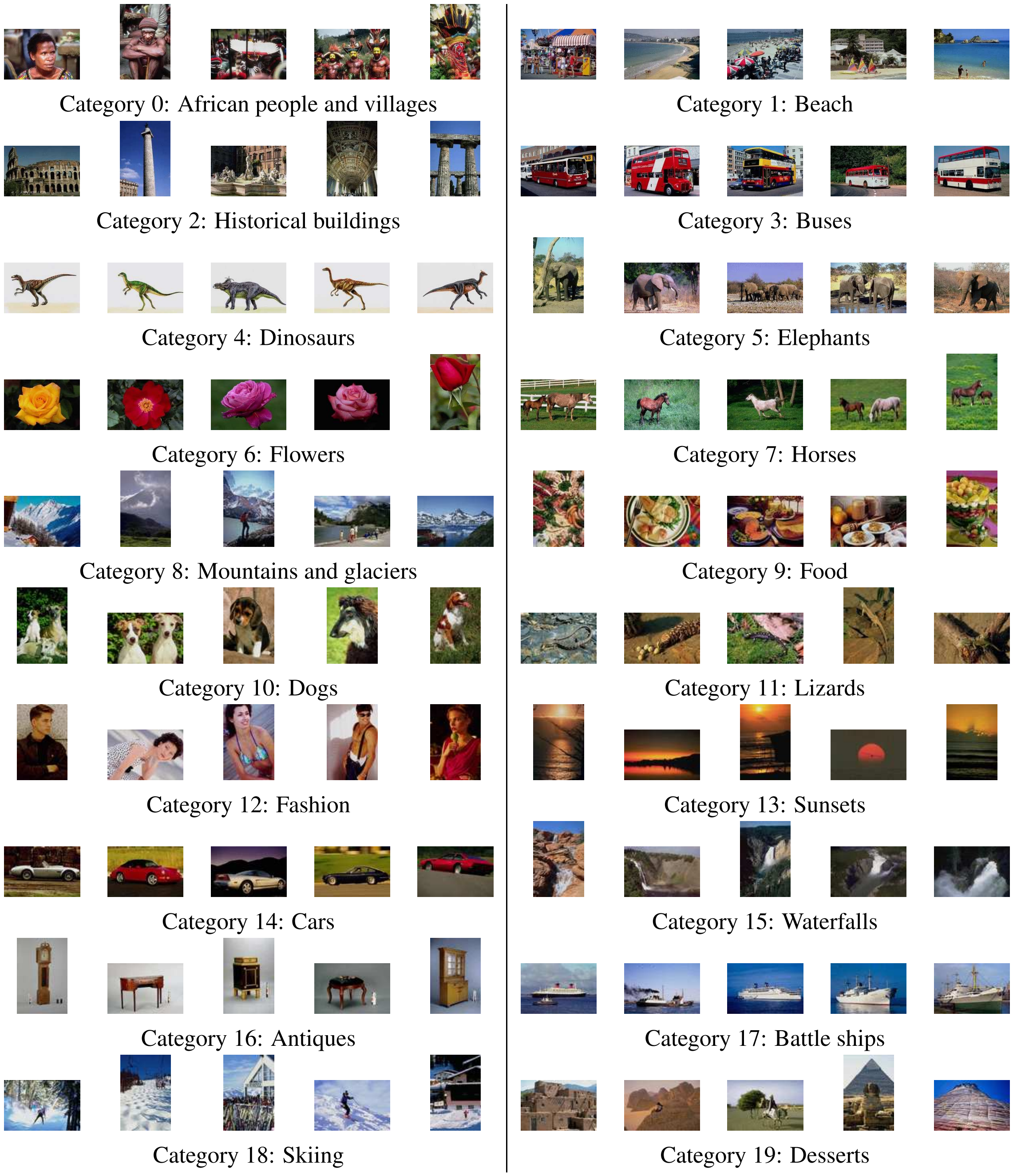

Figure 3: Sample images taken from 20 image categories.

4.2 Categorization Results

The classification results provided in Table 1 are based on images in Category 0 to Category 9, i.e., 1,000 images. Results for the whole data set will be given in Section 4.4. DD-SVM performs much better than Hist-SVM with a 14.8% difference in average classification accuracy. Compared with MI-SVM, the average accuracy of DD-SVM is 6.8% higher. As we will see in Section 4.4,

	Average Accuracy : [95% confidence interval]
DD-SVM	81.5% : [78.5%, 84.5%]
Hist-SVM	66.7% : [64.5%, 68.9%]
MI-SVM	74.7% : [74.1%, 75.3%]

Table 1: Image categorization performance of DD-SVM, Hist-SVM, and MI-SVM. The numbers listed are the average classification accuracies over 5 random test sets and the corresponding 95% confidence intervals. The images belong to Category 0 to Category 9. Training and test sets are of equal size.

	Cat. 0	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Cat. 7	Cat. 8	Cat. 9
Cat. 0	**67.7%**	3.7%	5.7%	0.0%	0.3%	8.7%	5.0%	1.3%	0.3%	7.3%
Cat. 1	1.0%	**68.4%**	4.3%	4.3%	0.0%	3.0%	1.3%	1.0%	15.0%	1.7%
Cat. 2	5.7%	5.0%	**74.3%**	2.0%	0.0%	3.3%	0.7%	0.0%	6.7%	2.3%
Cat. 3	0.3%	3.7%	1.7%	**90.3%**	0.0%	0.0%	0.0%	0.0%	1.3%	2.7%
Cat. 4	0.0%	0.0%	0.0%	0.0%	**99.7%**	0.0%	0.0%	0.0%	0.0%	0.3%
Cat. 5	5.7%	3.3%	6.3%	0.3%	0.0%	**76.0%**	0.7%	4.7%	2.3%	0.7%
Cat. 6	3.3%	0.0%	0.0%	0.0%	0.0%	1.7%	**88.3%**	2.3%	0.7%	3.7%
Cat. 7	2.3%	0.3%	0.0%	0.0%	0.0%	2.0%	1.0%	**93.4%**	0.7%	0.3%
Cat. 8	0.3%	15.7%	5.0%	1.0%	0.0%	4.3%	1.0%	0.7%	**70.3%**	1.7%
Cat. 9	3.3%	1.0%	0.0%	3.0%	0.7%	1.3%	1.0%	2.7%	0.0%	**87.0%**

Table 2: The confusion matrix of image categorization experiments (over 5 randomly generated test sets). Each row lists the average percentage of images (test images) in one category classified to each of the 10 categories by DD-SVM. Numbers on the diagonal show the classification accuracy for each category.

the difference becomes even greater as the number of categories increases. This suggests that the proposed method is more effective than MI-SVM in learning concepts of image categories under the same image representation. The MIL formulation of our method may be better suited for region-based image classification than that of MI-SVM.

Next, we make a closer analysis of the performance by looking at classification results on every category in terms of the confusion matrix. The results are listed in Table 2. Each row lists the average percentage of images in one category classified to each of the 10 categories by DD-SVM. The numbers on the diagonal show the classification accuracy for each category, and off-diagonal entries indicate classification errors. Ideally, one would expect the diagonal terms be all 1's, and the off-diagonal terms be all 0's. A detailed examination of the confusion matrix shows that two of the largest errors (the underlined numbers in Table 2) are errors between Category 1 (Beach) and Category 8 (Mountains and glaciers): 15.0% of "Beach" images are misclassified as "Mountains and glaciers;" 15.7% of "Mountains and glaciers" images are misclassified as "Beach." Figure 4 presents 12 misclassified images (in at least one experiment) from both categories. All "Beach" images in Figure 4 contain mountains or mountain-like regions, while all the "Mountains and glaciers" images

Figure 4: Some sample images taken from two categories: “Beach” and “Mountains and glaciers.” All the listed “Beach” images are misclassified as “Mountains and glaciers,” while the listed “Mountains and glaciers” images are misclassified as “Beach.”

have regions corresponding to river, lake, or even ocean. In other words, although these two image categories do not share annotation words, they are semantically related and visually similar. This may be the reason for the classification errors.

4.3 Sensitivity to Image Segmentation

Because image segmentation cannot be perfect, being robust to segmentation-related uncertainties becomes a critical performance index for a region-based image classification method. Figure 5 shows two images, “African people” and “Horses,” and the segmentation results with different numbers of regions (the results are obtained by varying the stopping criteria of the *k*-mean segmentation algorithm presented in Section 2). Regions are shown in their representative colors. We can see from Figure 5 that, under some stopping criteria, objects totally different in semantics may be clustered into the same region (under-segmented). While under some other stopping criteria, one object may be divided into several regions (over-segmented).

In this section, we compare the performance of DD-SVM with MI-SVM when the coarseness of image segmentation varies. To give a fair comparison, we control the coarseness of image segmentation by adjusting the stopping criteria of the *k*-means segmentation algorithm. We pick 5 different stopping criteria. The corresponding average numbers of regions per image (computed over $1,000$ images from Category 0 to Category 9) are 4.31, 6.32, 8.64, 11.62, and 12.25. The average classification accuracies (over 5 randomly generated test sets) under each coarseness level and the corresponding 95% confidence intervals are presented in Figure 6.

The results in Figure 6 indicate that DD-SVM outperforms MI-SVM on all 5 coarseness levels. In addition, for DD-SVM, there are no significant changes in the average classification accuracy for different coarseness levels. While the performance of MI-SVM degrades as the average number of regions per image increases. The difference in average classification accuracies between the two methods are 6.8%, 9.5%, 11.7%, 13.8%, and 27.4% as the average number of regions per image increases. This appears to support the claim that DD-SVM has low sensitivity to image segmentation.

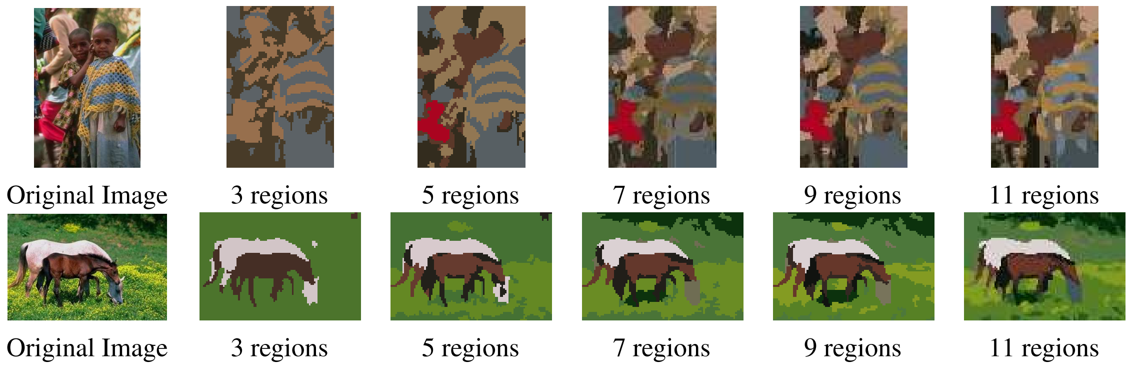

Figure 5: Segmentation results given by the k-means clustering algorithm with 5 different stopping criteria. Original images, which are taken from "African people" and "Horses" categories, are in the first column. Segmented regions are shown in their representative colors.

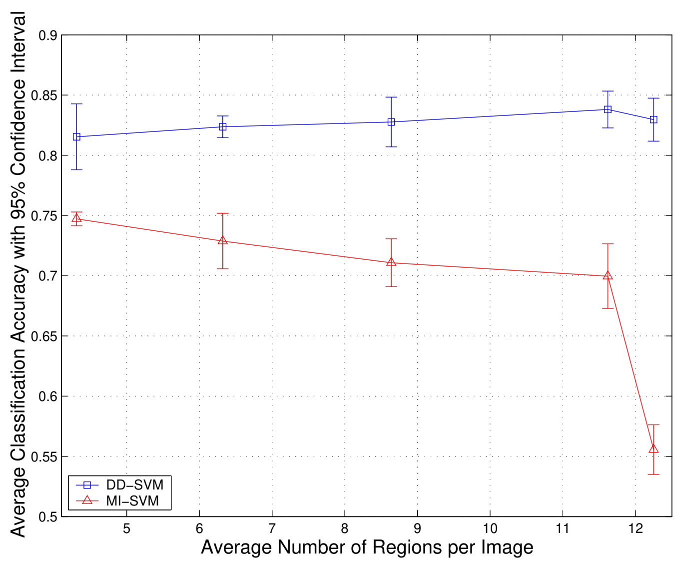

Figure 6: Comparing DD-SVM with MI-SVM on the robustness to image segmentation. The experiment is performed on $1,000$ images in Category 0 to Category 9 (training and test sets are of equal size). The average classification accuracies and the corresponding 95% confidence intervals are computed over 5 randomly generated test sets. The average numbers of regions per image are 4.31, 6.32, 8.64, 11.62, and 12.25.

4.4 Sensitivity to the Number of Categories in a Data Set

Although the experimental results in Section 4.2 and 4.3 demonstrate the good performance of DD-SVM using $1,000$ images in Category 0 to Category 9, the scalability of the method remains a question: how does the performance scale as the number of categories in a data set increases? We attempt to empirically answer this question by performing image categorization experiments over

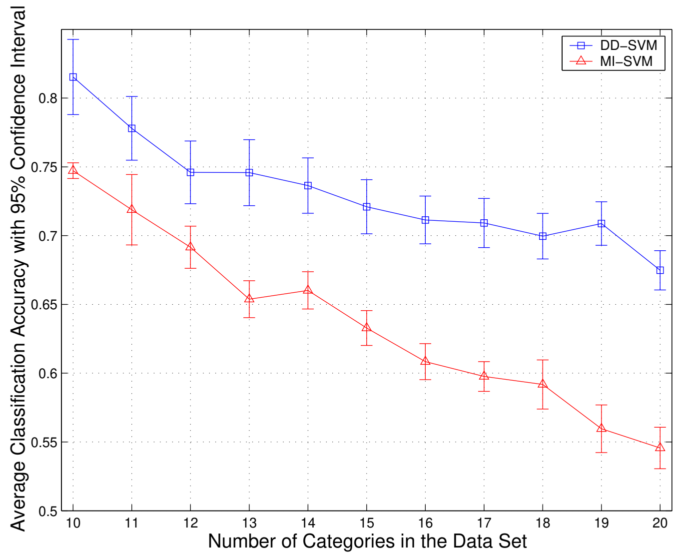

Figure 7: Comparing DD-SVM with MI-SVM on the robustness to the number of categories in a data set. The experiment is performed on 11 different data sets. The number of categories in a data set varies from 10 to 20. A data set with i categories contains $100 \times i$ images from Category 0 to Category $i-1$ (training and test sets are of equal size). The average classification accuracies and the corresponding 95% confidence intervals are computed over 5 randomly generated test sets.

data sets with different numbers of categories. A total of 11 data sets are used in the experiments. The number of categories in a data set varies from 10 to 20. A data set with i categories contains $100 \times i$ images from Category 0 to Category $i-1$. The average classification accuracies over 5 randomly generated test sets and the corresponding 95% confidence intervals are presented in Figure 7. We also include the results of MI-SVM for comparison.

We observe a decrease in average classification accuracy as the number of categories increases. When the number of categories becomes doubled (increasing from 10 to 20 categories), the average classification accuracy of DD-SVM drops from 81.5% to 67.5%. However, DD-SVM seems to be less sensitive to the number of categories in a data set than MI-SVM. This is indicated, in Figure 8, by the difference in average classification accuracies between the two methods as the number of categories in a data set increases. It should be clear that our method outperforms MI-SVM consistently. And the performance discrepancy increases with the number of categories. For the 1000-image data set with 10 categories, the difference is 6.8%. This number is nearly doubled (12.9%) when the number of categories becomes 20. In other words, the performance degradation of DD-SVM is slower than that of MI-SVM as the number of categories increases.

4.5 Sensitivity to the Size and Diversity of Training Images

We test the sensitivity of DD-SVM to the size of training set using $1,000$ images from Category 0 to Category 9 with the size of the training sets being 100, 200, 300, 400, and 500 (the number of images from each category is $\frac{\textit{size of the training set}}{10}$). The corresponding numbers of test images are 900,

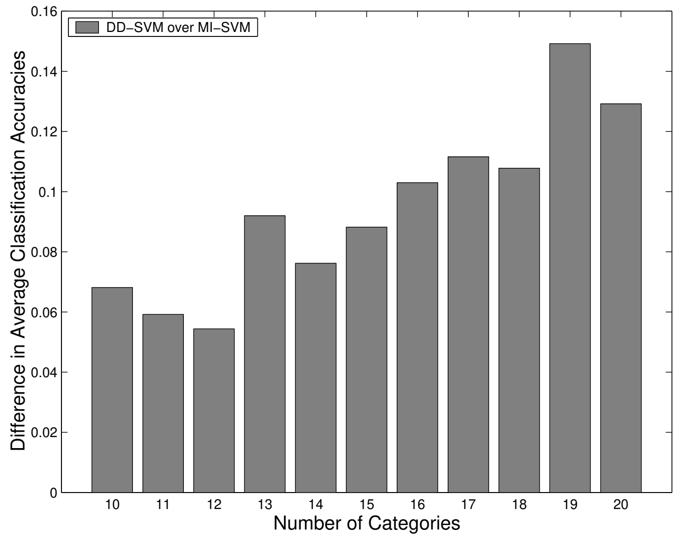

Figure 8: Difference in average classification accuracies between DD-SVM and MI-SVM as the number of categories varies. A positive number indicates that DD-SVM has higher average classification accuracy.

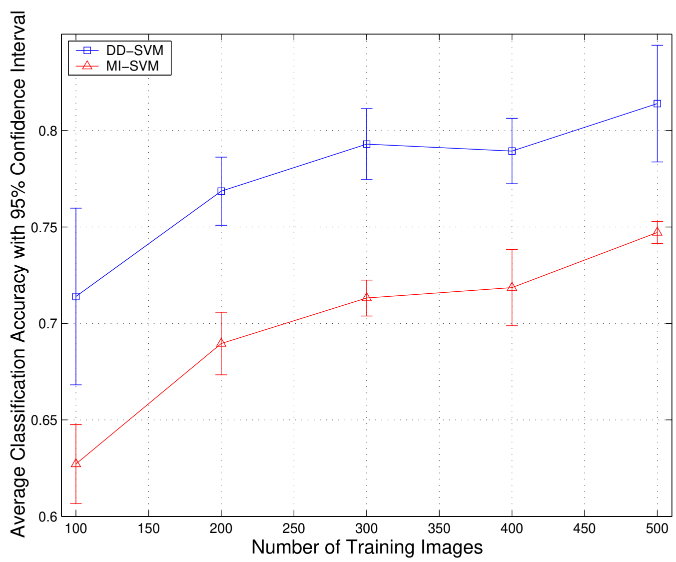

Figure 9: Comparing DD-SVM with MI-SVM as the number of training images varies from 100 to 500. The experiment is performed on $1,000$ images in Category 0 to Category 9. The average classification accuracies and the corresponding 95% confidence intervals are computed over 5 randomly generated test sets.

800, 700, 600, and 500. As indicated in Figure 9, when the number of training images decreases, the average classification accuracy of DD-SVM degrades as expected. Figure 9 also shows that the

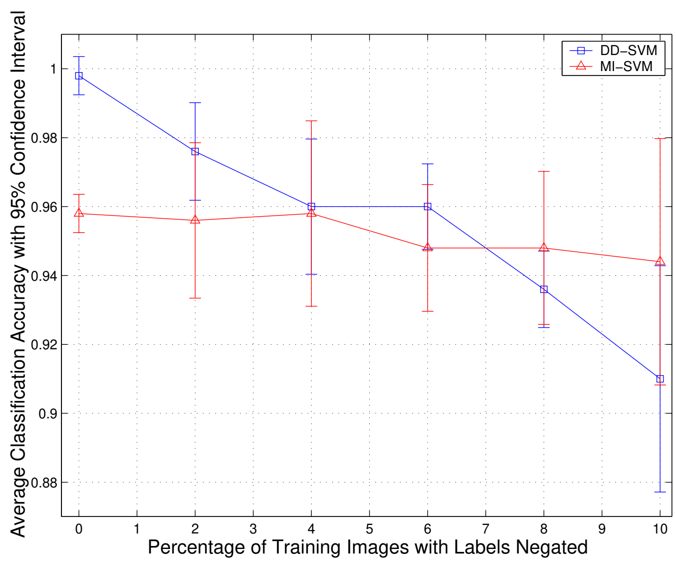

Figure 10: Comparing DD-SVM with MI-SVM as the diversity of training images varies. The experiment is performed on 200 images in Category 2 (Historical buildings) and Category 7 (Horses). The average classification accuracies and the corresponding 95% confidence intervals are computed over 5 randomly generated test sets. Training and test sets have equal size.

performance of DD-SVM degrades in roughly the same speed as that of MI-SVM: the differences in average classification accuracies between DD-SVM and MI-SVM are 8.7%, 7.9%, 8.0%, 7.1%, and 6.8% when the training sample size varies from 100 to 500.

To test the performance of DD-SVM as the diversity of training images varies, we need to define a measure of the diversity. In terms of binary classification, we define the diversity of images as a measure of the number of positive images that are "similar" to negative images and the number of negative images that are "similar" to positive images. In this experiment, training sets with different diversities are generated as follows. We first randomly pick $d\%$ of positive images and $d\%$ of negative images from a training set. Then, we modify the labels of the selected images by negating their labels, i.e., positive (negative) images become negative (positive) images. Finally, we put these images with new labels back to the training set. The new training set has $d\%$ of images with negated labels. It should be clear that $d = 0$ and $d = 50$ correspond to the lowest and highest diversities, respectively.

We compare DD-SVM with MI-SVM for $d = 0$, 2, 4, 6, 8, and 10 based on 200 images from Category 2 (Historical buildings) and Category 7 (Horses). The training and test sets have equal size. The average classification accuracies (over 5 randomly generated test sets) and the corresponding 95% confidence intervals are presented in Figure 10. We observe that the average classification accuracy of DD-SVM is about 4% higher than that of MI-SVM when $d = 0$. And this difference is statistically significant. However, if we randomly negate the labels of one positive image and one negative image in the training set (i.e., $d = 2$ in this experimental setup), the performance of DD-SVM is roughly the same as that of MI-SVM: although DD-SVM still leads MI-SVM by 2% of

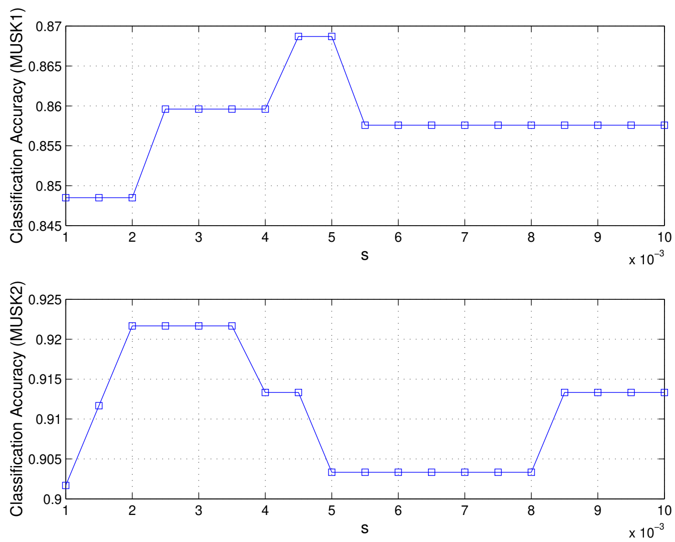

Figure 11: Average accuracy of 10-fold cross-validation on MUSK data sets using DD-SVM. The parameters are $C = 1,000$ and s taking 19 values evenly distributed in $[0.001, 0.01]$.

average classification accuracy, the difference is statistically-indistinguishable. As d increases, DD-SVM and MI-SVM generate roughly the same performance. This suggests that DD-SVM is more sensitive to the diversity of training images than MI-SVM. We attempt to explain this observation as follows. The DD function (3) used in Algorithm 3.1 is very sensitive to instances in negative bags. It is not difficult to derive from (3) that the DD value at a point is substantially reduced is there is a single instance from negative bags close to the point. Therefore, negating labels of one positive and one negative image could significantly modify the DD function, and consequently, the instance prototypes learned by Algorithm 3.1.

4.6 MUSK Data Sets

The MUSK data sets, MUSK1 and MUSK2 (Blake and Merz, 1998), are benchmark data sets for MIL. Both data sets consist of descriptions of molecules. Specifically, a bag represents a molecule. Instances in a bag represent low-energy conformations of the molecule. Each instance (or conformation) is defined by a 166-dimensional feature vector describing the surface of a low-energy conformation. The data were preprocessed by dividing each feature value by 100. This was done so that learning of instance prototypes would not begin at a flat area of the instance space. MUSK1 has 92 molecules (bags), of which 47 are labeled positive, with an average of 5.17 conformations (instances) per molecule. MUSK2 has 102 molecules, of which 39 are positive, with an average of 64.69 conformations per molecule.

Figure 11 shows the average accuracy of 10-fold cross-validation using DD-SVM with $C = 1,000$ and the s parameter of the Gaussian kernel taking the following values: 0.001, 0.0015, 0.002, 0.0025, 0.003, 0.0035, 0.004, 0.0045, 0.005 0.0055, 0.006, 0.0065, 0.007, 0.0075, 0.008, 0.0085, 0.009, 0.0095, 0.01. As a function of s, the average 10-fold cross-validation accuracy of DD-SVM varies within $[84.9\%, 86.9\%]$ (MUSK1) or $[90.2\%, 92.2\%]$ (MUSK2). For both data sets, the me-

	Average Accuracy	
	MUSK1	MUSK2
DD-SVM	85.8%	**91.3**%
IAPR	**92.4**%	89.2%
DD	88.9%	82.5%
EM-DD	84.8%	84.9%
MI-SVM	77.9%	84.3%
mi-SVM	87.4%	83.6%
MI-NN	88.0%	82.0%
Multinst	76.7%	84.0%

Table 3: Comparison of averaged 10-fold cross-validation accuracies on MUSK data sets.

dian of the average accuracy, which is robust over a range of parameter values, is reported in Table 3. Table 3 also summarizes the performance of seven MIL algorithms in the literature: IAPR (Dietterich et al., 1997), DD (Maron and Lozano-Pérez, 1998), EM-DD (Zhang and Goldman, 2002),[4] MI-SVM and mi-SVM (Andrews et al., 2003), MI-NN (Ramon and De Raedt, 2000), and Multinst (Auer, 1997). Although DD-SVM is outperformed by IAPR, DD, MI-NN, and mi-SVM on MUSK1, it generates the best performance on MUSK2. Overall, DD-SVM achieves very competitive accuracy values.

4.7 Speed

On average, the learning of each binary classifier using a training set of 500 images (4.31 regions per image) takes around 40 minutes of CPU time on a Pentium III 700MHz PC running the Linux operating system. Algorithm 3.1 is implemented in Matlab with the quasi-Newton search procedure written in the C programming language. Among this amount of time, the majority is spent on learning instance prototypes, in particular, the `FOR` loop of **LearnIPs($\mathcal{D}$)** in Algorithm 3.1. This is because the quasi-Newton search needs to be applied with every instance in every positive bag as starting points (each optimization only takes a few seconds). However, since these optimizations are independent of each other, they can be fully parallelized. Thus the training time may be reduced significantly.

5. Conclusions and Future Work

In this paper, we proposed a region-based image categorization method using an extension of Multiple-Instance Learning, DD-SVM. Each image is represented as a collection of regions obtained from image segmentation using the k-means algorithm. In DD-SVM, each image is mapped to a point in a bag feature space, which is defined by a set of instance prototypes learned with the Diverse Density function. SVM-based image classifiers are then trained in the bag feature space. We demonstrate that DD-SVM outperforms two other methods in classifying images from 20 distinct semantic classes. In addition, DD-SVM generates highly competitive results on the MUSK data sets, which are benchmark data sets for MIL.

4. The EM-DD results reported in Zhang and Goldman (2002) were obtained by selecting the optimal solution using the test data. The EM-DD result cited in this paper is provide by Andrews et al. (2003) using the correct algorithm.

The proposed image categorization method has several limitations:

- The semantic meaning of an instance prototype is usually unknown because the learning algorithm in Section 3 does not associate a linguistic label with each instance prototype. As a result, "region naming" (Barnard et al., 2003) is not supported by DD-SVM.
- It may not be possible to learn certain concepts through the method. For example, texture images can be designed using a simple object (or region), such as a T-shaped object. By varying orientation, frequency of appearance, and alignment of the object, one can get texture images that are visually different. In other words, the concept of texture depends on not only the individual object but also the spatial relationship of objects (or instances). But this spatial information is not exploited by the current work. As pointed out by one reviewer of the initial draft, a possible way to tackle this problem is to use Markov random field type of models (Modestino and Zhang, 1992).

The performance of image categorization may be improved in the following ways:

- The image segmentation algorithm may be improved. The current k-means algorithm is relatively simple and efficient. But over-segmentation and under-segmentation may happen frequently for a fixed stopping criterion. Although the empirical results in Section 4.3 show that the proposed method has low sensitivity to image segmentation, a semantically more accurate segmentation algorithm may improve the overall classification accuracy.
- The definition of the DD function may be improved. The current DD function, which is a multiplicative model, is very sensitive to instances in negative bags. It can be easily observed from (3) that the DD value at a point is significantly reduced if there is a single instance from a negative bag close to the point. This property may be desirable for some applications, such as drug discovery (Maron and Lozano-Pérez, 1998), where the goal is to learn a single point in the instance feature space with the maximum DD value from an almost "noise free" data set. But this is not a typical problem setting for region-based image categorization where data usually contain noise. Thus a more robust definition of DD, such as an additive model, may enhance the performance.

As pointed out by a reviewer of the initial draft, scene category can be a vector. For example, a scene can be $\{$*mountain*, *beach*$\}$ in one dimension, but also $\{$*winter*, *summer*$\}$ in the other dimension. Under this scenario, our current work can be applied in two ways: (a) design a multi-class classifier for each dimension, i.e., *mountain/beach* classifier for one dimension and *winter/summer* classifier for the other dimension; or (b) design one multi-class classifier taking all scene categories into consideration, i.e., *mountain-winter*, *mountain-summer*, *beach-winter*, and *beach-summer* categories.

In our experimental evaluations, image semantic categories are assumed to be well-defined. As pointed out by one of the reviewers, image semantics is inherently linguistic, therefore, can only be defined loosely. Thus a methodologically well-defined evaluation technique should take into account scenarios with differing amounts of knowledge about the image semantics. Unless this issue can be fully investigated, our image categorization results should be interpreted cautiously.

As continuations of this work, several directions may be pursued. The proposed method can potentially be applied to automatically index images using linguistic descriptions. It can also be integrated to content-based image retrieval systems to group images into semantically meaningful categories so that semantically-adaptive searching methods applicable to each category can be applied. The current instance prototype learning scheme may be improved by boosting techniques. Art and biomedical images would be interesting applications.

Acknowledgments

The material is based upon work supported by the National Science Foundation under Grant No. IIS-0219272 and CNS-0202007, The Pennsylvania State University, University of New Orleans, The Research Institute for Children, the PNC Foundation, SUN Microsystems under Grant EDUD-7824-010456-US, and NASA/EPSCoR DART Grant NCC5-573. The authors would like to thank Jia Li for making many suggestions on the initial manuscript. We thank the reviewers for valuable suggestions. We would also like to thank Jinbo Bi, Seth Pincus, Andrés Castaño, C. Lee Giles, Donald Richards, and John Yen for helpful discussions.

References

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems 15*, pages 561–568. Cambridge, MA:MIT Press, 2003.

P. Auer. On learning from mult-instance examples: empirical evaluation of a theoretical approach. In *Proc. 14th Int'l Conf. on Machine Learning*, pages 21–29, 1997.

K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

K. Barnard and D. Forsyth. Learning the semantics of words and pictures. In *Proc. 8th Int'l Conf. on Computer Vision*, pages II:408–415, 2001.

C. L. Blake and C. J. Merz. UCI Repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

A. Blum and A. Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30 (1):23–29, 1998.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, 2002.

O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5):1055–1064, 1999.

Y. Chen and J. Z. Wang. A region-based fuzzy feature matching approach to content-based image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1252–1267, 2002.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

I. Daubechies. *Ten Lectures on Wavelets*. Capital City Press, 1992.

T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2002.

Y. Gdalyahu and D. Weinshall. Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1312–1328, 1999.

A. Gersho. Asymptotically optimum block quantization. *IEEE Transactions on Information Theory*, 25(4):373–380, 1979.

M. M. Gorkani and R. W. Picard. Texture orientation for sorting photos 'at a glance'. In *Proc. 12th Int'l Conf. on Pattern Recognition*, pages I:459–464, 1994.

J. A. Hartigan and M. A. Wong. Algorithm AS136: A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

J. Huang, S. R. Kumar, and R. Zabih. An automatic hierarchical image classification scheme. In *Proc. 6th ACM Int'l Conf. on Multimedia*, pages 219–228, 1998.

T. Joachims. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. Edited by B. Schölkopf, C. J.C. Burges, and A.J. Smola, Cambridge, MA: MIT Press, 1999.

J. Li and J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1075–1088, 2003.

W. Y. Ma and B. Manjunath. NeTra: A toolbox for navigating large image databases. In *Proc. IEEE Int'l Conf. on Image Processing*, pages 568–571, 1997.

O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems 10*, pages 570–576. Cambridge, MA: MIT Press, 1998.

O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *Proc. 15th Int'l Conf. on Machine Learning*, pages 341–349, 1998.

D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W H Freeman & Co., 1983.

J. W. Modestino and J. Zhang. A Markov random field model-based approach to image interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(6):606–615, 1992.

K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees: a graphical model relating features, objects, and scenes. In *Advances in Neural Information Processing Systems 16*. Cambridge, MA:MIT Press, 2004.

S. A. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The art of scientific computing*. second edition, Cambridge University Press, New York, 1992.

J. Ramon and L. De Raedt. Multi instance neural networks. In *Proc. ICML-2000 Workshop on Attribute-Value and Relational Learning*, 2000.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

J. R. Smith and C.-S. Li. Image classification and querying using composite region templates. *Int'l J. Computer Vision and Image Understanding*, 75(1/2):165–174, 1999.

T. M. Strat. *Natural Object Recognition*. Berlin: Springer-Verlag, 1992.

M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *Proc. IEEE Int'l Workshop on Content-Based Access of Image and Video Databases*, pages 42–51, 1998.

M. Unser. Texture classification and segmentation using wavelet frames. *IEEE Transactions on Image Processing*, 4(11):1549–1560, 1995.

A. Vailaya, M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang. Image classification for content-based indexing. *IEEE Transactions on Image Processing*, 10(1):117–130, 2001.

N. Vasconcelos and A. Lippman. A Bayesian framework for semantic content characterization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 566–571, 1998.

J. Z. Wang, J. Li, R. M. Gray, and G. Wiederhold. Unsupervised multiresolution segmentation for images with low depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):85–91, 2001a.

J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):947–963, 2001b.

C. Yang and T. Lozano-Pérez. Image database retrieval with multiple-instance learning techniques. In *Proc. IEEE Int'l Conf. on Data Engineering*, pages 233–243, 2000.

H. Yu and W. Wolf. Scenic classification methods for image and video databases. In *Proc. SPIE Int'l Conf. on Digital Image Storage and archiving systems*, pages 2606:363–371, 1995.

Q. Zhang and S. A. Goldman. EM-DD: An improved multiple-instance learning technique. In *Advances in Neural Information Processing Systems 14*, pages 1073–1080. Cambridge, MA: MIT Press, 2002.

Q. Zhang, S. A. Goldman, W. Yu, and J. Fritts. Content-based image retrieval using multiple-instance learning. In *Proc. 19th Int'l Conf. on Machine Learning*, pages 682–689, 2002.

S. C. Zhu and A. Yuille. Region competition: unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996.