SVM-Based Generalized Multiple-Instance Learning via Approximate Box Counting

Qingping Tao QTAO@CSE.UNL.EDU
Stephen Scott SSCOTT@CSE.UNL.EDU
N. V. Vinodchandran VINOD@CSE.UNL.EDU
Thomas Takeo Osugi TOSUGI@CSE.UNL.EDU
Department of Computer Science & Engineering, University of Nebraska, Lincoln, NE 68588-0115, USA

Abstract

The multiple-instance learning (MIL) model has been very successful in application areas such as drug discovery and content-based image-retrieval. Recently, a generalization of this model and an algorithm for this generalization were introduced, showing significant advantages over the conventional MIL model in certain application areas. Unfortunately, this algorithm is inherently inefficient, preventing scaling to high dimensions. We reformulate this algorithm using a kernel for a support vector machine, reducing its time complexity from exponential to polynomial. Computing the kernel is equivalent to counting the number of axis-parallel boxes in a discrete, bounded space that contain at least one point from each of two multisets P and Q. We show that this problem is #P-complete, but then give a fully polynomial randomized approximation scheme (FPRAS) for it. Finally, we empirically evaluate our kernel.

1. Introduction

Dietterich et al. (1997) introduced the multiple-instance learning (MIL) model motivated by the problem of predicting whether a molecule would bind at a particular site. Since shape of a molecule largely determines binding affinity, they represented each molecule by a high-dimensional vector that describes its shape, and labeled molecules that bind at a site as positive examples and those that do not bind as negative. Then they learned an axis-parallel box that distinguishes the positives from the negatives. The motivation for the MIL model is the fact that a single molecule can have multiple conformations (shapes), and only one conformation need bind at the site for the molecule to be considered positive. Thus when an example is negative, all conformations in it are negative, but if an example is positive, then it may be the case that only one conformation of the set is positive, and the learner does not know which one. Since its introduction, the MIL model has been applied to content-based image retrieval (Maron & Ratan, 1998; Zhang et al., 2002), where each instance in a multi-instance example (*bag*) represents a feature of an image, and it is not known which feature corresponds to the content the user wants to retrieve. As with binding prediction, the MIL model used for content-based image retrieval assumes that the label of an example is a disjunction of the labels of the instances in the example.

Recently, Scott et al. (2003) generalized the MIL model, allowing an example's label to be represented as an r-of-k threshold function rather than as a disjunction. They then presented an algorithm (referred to here as GMIL-1) for learning general geometric concepts in this new model and evaluated it empirically on problems from robot vision, content-based image retrieval, binding affinity, and biological sequence analysis. In all experiments, GMIL-1 was competitive with algorithms from the conventional MIL model. Further, on problems requiring the labeling function to be more general than a disjunction, GMIL-1 showed a significant advantage in prediction error.

GMIL-1 works by first explicitly enumerating all axis-parallel boxes in the space $\{0,\ldots,s\}^d$, where d is the number of dimensions and s is the number of discrete values in each dimension. Then it assigns boolean attributes to these boxes, and gives these attributes to Littlestone's (1991) algorithm Winnow, which learns a linear threshold unit. The time complexity of this algorithm is exponential in d, which obviously limits the applicability of GMIL-1. While there has been progress in developing heuristics to significantly speed up this algorithm in practice (Tao & Scott, 2004), the algorithm is still limited in its scalability.

Appearing in *Proceedings of the 21^{st} International Conference on Machine Learning*, Banff, Canada, 2004. Copyright 2004 by the authors.

We show that a kernel exists that exactly corresponds to the feature mapping used by GMIL-1. To compute the kernel, one takes two bags of points P and Q and counts the number of boxes defined on $\{0,\ldots,s\}^d$ that contain at least one point from P and at least one point from Q. We first show that this problem is #P-complete, present a fully polynomial randomized approximation scheme (FPRAS) for it, and then empirically evaluate our kernel.

The rest of this paper is organized as follows. In the next section we introduce some notation. In Section 3 we describe the MIL model and present Scott et al.'s generalization of it, as well as their algorithm GMIL-1. Then in Section 4 we present our kernel-based reformulation of GMIL-1. We show that computing this kernel is equivalent to counting the number of boxes that contain at least one point from both sets P and Q, a problem that we formally define in Section 5 as #BOXAnd. We prove that this problem is #P-complete and give an FPRAS for it. In Section 6 we describe experimental results of our new kernel on applications such as content-based image retrieval, prediction of drug affinity to bind to multiple sites simultaneously, protein sequence identification, and the Musk data sets. Finally, we conclude in Section 7.

2. Notation and Definitions

Let $\mathcal{X}$ denote $\{0,\ldots,s\}^d$ (though our results trivially generalize to $\mathcal{X} = \prod_{i=1}^d \{0,\ldots,s_i\}$). Let $B_{\mathcal{X}}$ denote the set of all axis-parallel boxes (including degenerate boxes) from $\mathcal{X}$. We uniquely identify any box $b \in B_{\mathcal{X}}$ as a pair (b_ℓ, b_u), where b_ℓ is the "lower left" corner and b_u is the "upper right" corner. There are $s+1$ possible values for each corner, so the number of intervals in each dimension is $\binom{s+1}{2} + s + 1$ since we allow degenerate intervals. Thus $|B_{\mathcal{X}}| = \left(\binom{s+1}{2} + s + 1\right)^d = \binom{s+2}{2}^d$.

For multisets $P, Q \subseteq \mathcal{X}$, let $B(P)$ denote the set of boxes in $B_{\mathcal{X}}$ that contain a point from P and $B(P \wedge Q)$ denote the set of boxes in $B_{\mathcal{X}}$ that contain a point from P and a point from Q. When P and Q contain single points then we will omit set notation. For example, $B(\{p\} \wedge \{q\})$ will be denoted as $B(p \wedge q)$.

We will use vector notation to refer to points in $\mathcal{X}$ only when it is necessary (e.g. in Section 5.1); otherwise we will just use lower case letters to refer to points in $\mathcal{X}$. The notion of approximation that we use is defined as follows.

Definition 1 *Let f be a counting problem. Then a randomized algorithm $\mathcal{A}$ is an* FPRAS (Fully Polynomial Randomized Approximation Scheme) *if for any instance x, and parameters $\epsilon, \delta > 0$,*

$$\Pr\left[|\mathcal{A}(x) - f(x)| \le \epsilon f(x)\right] \ge 1 - \delta$$

*and A's running time is polynomial in $|x|$, $1/\epsilon$, and $1/\delta$. Further, we call $\mathcal{A}(x)$ an ϵ-*good approximation *of $f(x)$.*

3. Multiple-Instance Learning

In the original MIL model (Dietterich et al., 1997), each example P is a *bag* (multiset) of instances, and P is given a label of positive if and only if at least one of the instances in P is labeled positive (it is unknown which instance(s) in P are labeled positive). Typically, the label of a point $p \in P$ is determined by its proximity to a target point c. Since its introduction, the MIL model has been extensively studied (Wang & Zucker, 2000; Andrews et al., 2002; Dooly et al., 2002; Ray & Page, 2001; Maron & Lozano-Pérez, 1998; Zhang & Goldman, 2001) with applications focusing on molecular binding affinity (related to drug discovery) and content-based image retrieval.

Scott et al. (2003) generalized the MIL model such that rather than P's label being a disjunction of the labels of the instances in P, the label is represented by a threshold function. In contrast to the conventional MIL model, in their model the target concept is defined by two *sets* of points. Specifically, they defined their concepts by a set of k "attraction" points $C = \{c_1,\ldots,c_k\}$ and a set of k' "repulsion" points $\bar{C} = \{\bar{c}_1,\ldots,\bar{c}_{k'}\}$. Then the label for a bag $P = \{p_1,\ldots,p_n\}$ is positive if and only if there is a subset of r points $C' \subseteq C \cup \bar{C}$ such that each attraction point $c_i \in C'$ is near some point in P (where "near" is defined as within a certain distance under some weighted norm) and each repulsion point $\bar{c}_j \in C'$ is not near any point in P.

In other words, if one defines a boolean attribute a_i for each attraction point $c_i \in C$ that is 1 if there exists a point $p \in P$ near it and 0 otherwise and another boolean attribute $\bar{a}_i$ for each repulsion point $\bar{c}_j \in \bar{C}$ that is 1 if there is no point from P near it, then P's label is an r-of-$(k+k')$ threshold function over the attributes. Note that if $r = 1$ and if there are no repulsion points, then this model is the conventional multi-instance model, except that there are multiple target points and the final concept is a union of these points.

Independently of Scott et al., Weidmann et al. (2003) defined their own generalizations of the MIL model. The first (*presence-based* MIL) is the same as Scott et al.'s model with $r = k$ and no repulsion points. Their second (*threshold-based* MIL) generalizes presence-based MIL by requiring each $c_i \in C$ to be near at least t_i distinct points from P for P to be labeled positive, where t_i is a nonnegative integer that is part of the definition of the target concept. Their third model (*count-based* MIL) generalizes threshold-based by requiring the number of distinct points from P that are near c_i to be at least t_i and at most z_i.

Count-based MIL can represent the idea of repulsion points by setting $z_i = 0$ for each repulsion point. Thus this model generalizes the one of Scott et al. when $r = k + k'$. However, the ability of Scott et al.'s model to represent r-of-$(k + k')$ threshold concepts for $r < k + k'$ expands its representational ability beyond the scope of the generalizations of Weidmann et al. (2003). As an example of why this is useful, consider the representation of a human face with shape-based features. For the eyes, the target concept might require two regions with elongation near $3/2$ and an Euler number (number of connected regions minus number of holes) of 0. In addition, one wants (for the mouth) one region with elongation near 8 and 0 or 1 holes in the region. Then there are other constraints (in terms of the above features or based on other shape descriptors) for the shape of the face. There exists a target concept for this case in presence-based MIL so long as all features are visible. But if some parts of the face are occluded (e.g. due to the subject wearing sunglasses) and the set of parts that are occluded can vary, then it is difficult to represent the target concept with even count-based MIL. In contrast, an r-of-k threshold function like that from Scott et al. is a natural way to represent the target concept.

When they introduced their generalized MIL model, Scott et al. also gave an algorithm (GMIL-1) for it. GMIL-1 is adapted from an algorithm by Goldman et al. (2001). GMIL-1 learns geometric concepts, and Scott et al. applied it to various application areas: robot vision, content-based image retrieval, biological sequence analysis, and molecular binding. In all tests, GMIL-1 was competitive with (and often superior to) the MIL algorithms Diverse Density (Maron & Lozano-Pérez, 1998) and EMDD (Zhang & Goldman, 2001). GMIL-1's advantage was most clear when there was no way to represent a target concept in the original MIL model, such as a content-based image retrieval task in which the objective was to identify natural scenes containing a field but not a sky (see Section 6.1).

GMIL-1 can be summarized as follows. It operates in a discretized feature space (without loss of generality, assume it is $\mathcal{X} = \{0, \ldots, s\}^d$). GMIL-1 enumerates the set $B_{\mathcal{X}}$ of all possible boxes (including degenerate ones) in $\mathcal{X}$ (so $|B_{\mathcal{X}}| = \binom{s+2}{2}^d$) and creates an attribute a_b for each box $b \in B_{\mathcal{X}}$. Given a bag $P \in \mathcal{X}^n$, the algorithm sets $a_b = 1$ if some point from P lies in b and $a_b = 0$ otherwise. To capture the notion of repulsion points, they also defined complementary[1] attributes $\bar{a}_b = 1 - a_b$. These $N = 2|B_{\mathcal{X}}|$ attributes are given to the algorithm Winnow (Littlestone, 1991), which learns a linear threshold unit.

Winnow maintains a weight vector $\vec{w} \in \Re^{+N}$ (N-dimensional positive real space), initialized to all 1s. Upon receiving input $\vec{x}_i \in [0,1]^N$, Winnow makes its prediction $\hat{y}_i = +1$ if $\vec{w} \cdot \vec{x}_i \geq \theta$ and -1 otherwise ($\theta > 0$ is a threshold). Given the true label y_i, the weights are updated as follows: $\vec{w} = \vec{w}\,\alpha^{\vec{x}_i(y_i - \hat{y_i})/2}$ for some $\alpha > 1$. Thus Winnow is very similar to the Perceptron algorithm, but updates its weights multiplicatively rather than additively.

Using the above remapping of bags to boolean attributes, a target concept defined by some C, $\bar{C}$, and r can be represented by an r-of-$(k + k')$ threshold function over the attributes a_{c_i} for $c_i \in C$ and $\bar{a}_{\bar{c}_j}$ for $\bar{c}_j \in \bar{C}$. Such a function can easily be learned by Winnow while making only $O(r(k + k')d \log s)$ mistakes (Littlestone, 1991) in an on-line learning setting.

Unfortunately, the time complexity of this algorithm is linear in N, which is exponential in $\log s$ and d. By applying a grouping trick from Goldman et al. (2001), one can reduce the time complexity from $\Omega(s^{2d})$ to $O\left((nr(k + k')d \log s)^{2d}\right)$ for the on-line learning case and $O\left((nm)^{2d}\right)$ for the batch learning case, where m is the number of bags in the training set and n is the number of points in each bag. However, this improvement is insufficient to allow scaling of the algorithm to general d. Recently, Tao and Scott (2004) developed heuristics (GMIL-2) to significantly speed up this algorithm in practice, including a version that runs in time $poly(d)$ in exchange for being exponential in nm. However, both variations still have exponential time complexity and are thus limited to small nm or small d.

4. Kernel-Based Reformulation of GMIL-1

In seeking out an algorithm that scales polynomially in both n and d, we define a kernel that can be used with a support vector machine to efficiently learn geometric multiple-instance concepts. We will show that computing such a kernel on two bags P and Q corresponds to counting the number of boxes that contain at least one point from each of P and Q. After we show that this problem is #P-complete, we develop an FPRAS for it.

Observation 1 *Consider two bags $P, Q \subseteq \mathcal{X}$ and a mapping $\vec{\phi}_{\wedge}(P) = (a_1, \ldots, a_N)$ where $a_i = 1$ if the corresponding box $b_i \in B_{\mathcal{X}}$ contains a point from P and 0 otherwise. Then when using an SVM for learning, the remapping used by GMIL-1 corresponds to using the kernel*

$$k_{\wedge}(P, Q) = \vec{\phi}_{\wedge}(P) \cdot \vec{\phi}_{\wedge}(Q) = |B(P \wedge Q)| \ ,$$

where $B(P \wedge Q)$ is the set of boxes that contain a point from P and contain a point from Q.

Proof: Since $\vec{\phi}_{\wedge}(P)$ and $\vec{\phi}_{\wedge}(Q)$ are binary vectors, their dot product is simply the number of 1s in corresponding positions. Since a bit from $\vec{\phi}_{\wedge}(P)$ is 1 if and only if the corresponding box contains a point from P, the value of $k_{\wedge}(P, Q)$ is obviously $|B(P \wedge Q)|$. □

[1] This was done because Winnow in its standard form cannot represent negative weights. In our kernel formulation of Section 4, we only use the N attributes a_b.

5. The Box Counting Problem #BOXAnd

From Observation 1, it follows that the kernel $k_\wedge$ for GMIL-1 corresponds to the box counting problem that we call #BOXAnd, which we now define. The input to the problem is a triple $\langle \mathcal{X}, P, Q\rangle$. The problem #BOXAnd is to compute $|B(P \wedge Q)|$: the number of boxes in $B_{\mathcal{X}}$ that contain at least one point from each of P and Q. In this section we prove that #BOXAnd is #P-complete, and then we present an FPRAS for it.

5.1. Hardness Result for #BOXAnd

It is easy to see that #BOXAnd is in #P: given a particular box $b \in B_{\mathcal{X}}$, it is simple to verify that b contains a point from both P and Q. We now prove #P-completeness by reducing from the *monotone DNF counting problem* (#MDNF), shown to be #P-complete by Valiant (1979). An instance of #MDNF is a monotone boolean formula F (i.e. with no negated literals) in disjunctive normal form, and an algorithm for this problem is to output the number of satisfying assignments of F.

Let F be a monotone DNF formula in n variables with m monotone terms $t_1, t_2, \ldots, t_m$. Let $S(F)$ denote the set of all satisfying assignments of F. Then $S(F) = \bigcup_i S(t_i)$. Each monotone term t can be identified with an n-bit binary vector $\vec{v_t}$ as follows: $\vec{v_t} = v_1 v_2 \ldots v_n$ where $v_i = 1$ if $x_i \in t$ and $v_i = 0$ if $x_i \notin t$. Then, since t is monotone, the set of satisfying assignments for t, $S(t) = \{\vec{a} \mid \vec{a} \geq \vec{v_t}\}$. (For two n-bit vectors $\vec{u} = (u_1, u_2, \ldots, u_n)$ and $\vec{v} = (v_1, v_2, \ldots, v_n)$, $\vec{u} \geq \vec{v}$ iff $u_i \geq v_i$ for all $1 \leq i \leq n$.)

Theorem 2 #BOXAnd is #P-complete.

Proof: We have already established that #BOXAnd is in #P. We now show that #BOXAnd is #P-hard by reducing #MDNF to a special case of #BOXAnd where $\mathcal{X} = H_n = \{0,1\}^n$. The reduction f takes a formula $F = \bigvee_{1 \leq i \leq m} t_i$ and maps it to an instance $f(F) = \langle H_n, P, Q\rangle$ where $P = \{\vec{0}\}$ and $Q = \{\vec{v}_{t_1}, \vec{v}_{t_2}, \ldots, \vec{v}_{t_m}\}$.

We now argue that $|S(F)|$ equals the number of solutions to $\langle H_n, P, Q\rangle$ of #BOXAnd. Clearly, $B(\vec{0} \wedge \vec{v}) = \{(\vec{0}, \vec{u}) \mid \vec{u} \geq \vec{v}\}$. For any term t_i, $\vec{a} \in S(t_i) \Leftrightarrow \vec{a} \geq \vec{v}_{t_i} \Leftrightarrow (\vec{0}, \vec{a}) \in B(\vec{0} \wedge \vec{v}_{t_i})$. Thus the number of satisfying assignments of F = $|\bigcup_{1 \leq i \leq m} S(t_i)| = |\bigcup_{1 \leq i \leq m} B(\vec{0} \wedge \vec{v_{t_i}})| = |B(\{\vec{0}\} \wedge Q)|$ = the number of solutions to $\langle H_n, P, Q\rangle$. □

5.2. An FPRAS for #BOXAnd

Our algorithm for estimating $|B(P \wedge Q)|$ is based on the general technique from Karp et al. (1989) on the union of sets problem. In this problem, the goal is to take a description of m sets $B_1, \ldots, B_m$ and estimate the size of $B = \bigcup_{i=1}^m B_i$. In order to apply their technique, three criteria must be satisfied. First, for all $i \in \{1, \ldots, m\}$, $|B_i|$ must be easily computed. Second, for all $i \in \{1, \ldots, m\}$, we must be able to sample uniformly elements from B_i. Finally, given any $s \in B$ and any $i \in \{1, \ldots, m\}$, we must be able to easily determine if $s \in B_i$.

If the above criteria are satisfied, Karp et al.'s algorithm proceeds as follows. First define $U = \{(s,i) \mid s \in B_i \text{ and } 1 \leq i \leq m\}$ (so $|U| = \sum_{i=1}^m |B_i|$). Define another set $G = \{(s,i) \mid i \text{ is the smallest index such that } s \in B_i\}$. Then we have $G \subseteq U$ and $|G| = |B|$. Karp et al.'s algorithm runs in trials. For each trial, first a set B_i is chosen at random with probability $|B_i|/|U|$. Then an element $s \in B_i$ is chosen uniformly at random. These two steps together uniformly sample a pair (s,i) from U. Finally, if $(s,i) \in G$ we increment a counter γ, otherwise do nothing. The final estimate of $|B|$ is $|U|\gamma/S$, where S is the number of samples drawn. The following theorem bounds the error of this approximation.

Theorem 3 *(Karp et al., 1989) If* $S \geq 4(|U|/|G|)\ln(2/\delta)/\epsilon^2$*, then*

$$\Pr\left[(1-\epsilon)|B| \leq |U|\gamma/S \leq (1+\epsilon)|B|\right] \geq 1 - \delta.$$

We now apply Karp et al.'s result to #BOXAnd. Recall that for two points $p, q \in \mathcal{X}$, $B(p \wedge q)$ denotes the set of boxes that contain both p and q. Let $W = |B(P \wedge Q)|$. Then $W = |\bigcup_{p \in P, q \in Q} B(p \wedge q)|$. It is straightforward to compute $|B(p \wedge q)|$. Given points $p, q \in \mathcal{X}$, let $\ell = (\ell_1, \ldots, \ell_d)$ be the lower corner of the bounding box of p and q, i.e. $\ell_i = \min\{p_i, q_i\}$ for all i. Similarly define $u = (u_1, \ldots, u_d)$ as the upper corner. Then $|B(p \wedge q)| = \left(\prod_{1 \leq i \leq d}(\ell_i + 1)\right)\left(\prod_{1 \leq i \leq d}(s - u_i + 1)\right)$. Since we can exactly compute $|B(p \wedge q)|$ for all $(p,q) \in P \times Q$ and there are only n^2 such sets, we can easily choose a set $B(p \wedge q)$ with probability $|B(p \wedge q)| / \left(\sum_{p \in P, q \in Q} |B(p \wedge q)|\right)$. Further, since we can uniformly sample from $B(p \wedge q)$ by uniformly selecting lower and upper corners, we can uniformly sample from the set $U = \{(p,q,c) \mid p \in P, q \in Q, c \in B(p \wedge q)\}$.

Note that $|U| = \sum_{p \in P, q \in Q} |B(p \wedge q)|$. Now consider all the pairs (p,q) such that $p \in P$ and $q \in Q$. We define a total order $\prec$ on these pairs by sorting first by p's index in P, and then by q's index in Q. I.e. given points $p_i, p_{i'} \in P$ and $q_j, q_{j'} \in Q$, we define $(p_i, q_j) \prec (p_{i'}, q_{j'})$ iff $i < i'$ or $i = i'$ and $j < j'$.

Consider another set $G = \{(p,q,c) \in U \mid \text{there are no pairs } (p', q') \prec (p,q) \text{ such that } c \in B(p' \wedge q')\}$. Then $|G| = |\bigcup_{p \in P, q \in Q} B(p \wedge q)| = W$. We check whether $(p,q,c) \in G$ in $O(dn^2)$ time by checking c against each set $B(p \wedge q)$ for all $p \in P$ and $q \in Q$. Finally, we note that

$$|U| = \sum_{p \in P, q \in Q} |B(p \wedge q)| \leq n^2 \max_{p,q} |B(p \wedge q)| \leq n^2 |G| \quad . \tag{1}$$

Thus by drawing a sufficient number of samples (p, q, c) uniformly from U and incrementing γ when $(p, q, c) \in G$, we know that $\hat{W} = |U|\gamma/S$ is an ϵ-good approximation of W, as stated in the following theorem. Since S, the time to draw each sample, and the time to check each sample for membership in G are all polynomial in n, d, $\log s$, $1/\epsilon$, and $1/\delta$, our algorithm for #BOXAnd is an FPRAS.

Theorem 4 *If $S \geq 4n^2 \ln(2/\delta)/\epsilon^2$, then*

$$\Pr\left[(1-\epsilon)W \leq \hat{W} = |U|\gamma/S \leq (1+\epsilon)W\right] \geq 1 - \delta.$$

Proof: Directly from application of Equation (1) to Theorem 3. □

Our algorithm as presented has running time $O(n^4 d(\log s)\ln(1/\delta)/\epsilon^2)$ since it takes $O(dn^2)$ steps to check each sample for membership in G. However, it is possible to check for membership in G in time $O(dn)$. Given a triple (p_i, q_j, c) sampled from U, first check all points $p_{i'} \in P$ that are contained in c. If $i' < i$ for some $p_{i'} \in c$, then $(p_{i'}, q_j) \prec (p_i, q_j)$ and $(p_i, q_j, c) \notin G$. If there does not exist such a $p_{i'}$, then check all points $q_{j'} \in Q$ that are contained in c. Again, if $j' < j$ for some $q_{j'} \in c$, then $(p_i, q_{j'}) \prec (p_i, q_j)$ and $(p_i, q_j, c) \notin G$. If no such $q_{j'}$ exists, then $(p_i, q_j, c) \in G$. This check requires time $O(dn)$, reducing the total running time to $O(n^3 d(\log s)\ln(1/\delta)/\epsilon^2)$.

To further reduce time complexity, we can adapt Karp et al.'s "self-adjusting coverage algorithm" (a more efficient algorithm for the union of sets problem) to get an algorithm[2] for #BOXAnd with running time $O(n^2 d(\log s)\ln(1/\delta)/\epsilon^2)$.

5.3. Discussion

According to Observation 1, $k_\wedge(P, Q)$ is a kernel since it is the dot product of two remapped vectors. But there is no guarantee that the Gram matrix computed by our approximation algorithm is positive semidefinite. However, it is reasonable to believe that if ϵ is small and $k_\wedge$'s Gram matrix has no zero Eigenvalues, the approximated matrix would not adversely affect SVM optimization. In fact, in our experiments, our approximate kernel works very well when we set $\epsilon = 0.1$.

Another observation about our kernel is that its Gram matrix potentially can have large diagonal elements relative to the off-diagonal elements. $k_\wedge(P, Q)$ is the number of boxes that contain a point from P and a point from Q. If few points from P and Q are close to each other, $k_\wedge(P, Q)$ will be much smaller than $k_\wedge(P, P)$ and $k_\wedge(Q, Q)$. This worsens when d is large. For example, in our Musk experiments, the ratio of diagonal entries in the kernel matrix to the off-diagonal entries was often around 10^{50}. In practice, SVMs do not work well with diagonally dominated Gram matrices. To solve this problem, Schölkopf et al. (2002) propose first using a nonlinear function to reduce the value of each matrix element, such as a sub-polynomial function $\varphi(x) = \text{sign}(x) \cdot |x|^\rho$ with $0 < \rho < 1$. To then get a positive definite Gram matrix, they use the empirical kernel map $\phi_n(x) = (k'(x, x_1), k'(x, x_2), \cdots, k'(x, x_n))$, where $k'(x, x_i) = \varphi(k(x, x_i))$. Finally they apply the kernel $k_{emp}(x, y) = \phi_n(x) \cdot \phi_n(y)$. In the empirical kernel, the set $\{x_1, \cdots, x_n\}$ can consist of all training and testing bags (referred to as *transduction*) or of only the training bags. We applied this method with $k_\wedge$ to address our diagonal dominance problem (see Section 6.4). Also, since k_{emp} is always a kernel no matter what k is, with this method we do not need to worry about whether our ϵ-approximation of $k_\wedge$ is really a kernel.

6. Experimental Results

To compare our kernel to the algorithm GMIL-1 of Scott et al. (2003) and GMIL-2 of Tao and Scott (2004), we tested our kernel with SVM^{light} (Joachims, 1999) on a subset of the data sets they used: content-based image retrieval (real data) and predicting when drugs would bind at multiple sites of a molecule (simulated data). We also tested our kernel on the protein data used by Wang et al. (2004). All these data sets have dimension at most 8, since GMIL-1 and GMIL-2 cannot scale well to higher dimensions. To evaluate our algorithm on high-dimensional data, we also tested on simulated multi-site binding data and the Musk data sets from the UCI repository (Blake et al., 2004). In all our tests, we approximated $k_\wedge$ using our self-adjusting approximation algorithm with $\epsilon = 0.1$ and $\delta = 0.01$.

6.1. Content-Based Image Retrieval

In content-based image retrieval (CBIR), the user presents examples of desired images, and the task is to determine commonalities among the query images and retrieve similar ones from the database. Maron and Ratan (1998) explored the use of conventional MIL for CBIR. They filtered and subsampled the images and then extracted "blobs" (groups of m adjacent pixels), which were mapped to one point in a bag. Then they used the algorithm diverse density (DD) (Maron & Lozano-Pérez, 1998) to learn a hypothesis and find candidate images in the database. This work was later extended by Zhang et al. (2002).

[2] For brevity, we omit the details of the self-adjusting algorithm since it is similar to the one already presented.

Table 1. Generalization error for CBIR (top), protein (middle), and drug affinity (bottom) learning tasks.

Task	$k_\wedge$	GMIL-1	GMIL-2	EMDD	DD
sunset	0.088	0.095	0.098	0.096	0.099
conj.	0.108	0.134	0.147	0.215	0.181
protein	0.213	N/A	0.251	0.338	0.608
5-dim	0.205	0.212	0.218	0.191	0.196
10-dim	0.175	N/A	N/A	0.223	0.216
20-dim	0.207	N/A	N/A	0.268	0.255

We experimented with the two CBIR tasks used by Scott et al. One is the "sunset" task: to distinguish images containing sunsets from those not containing sunsets. Like Zhang et al., Scott et al. built 30 random testing sets of 720 examples (120 positives and 600 negatives): 150 negatives each from the waterfall, mountain, field, and flower sets. Each of 30 training sets consisted of 50 positives and 50 negatives.

Another CBIR task Scott et al. experimented with was to test a conjunctive CBIR concept, where the goal was to distinguish images containing a field with no sky from those containing a field and sky or containing no field. Zhang et al.'s field images that contained the sky were relabeled from positive to negative. Each training set had 6 bags of each of flower, mountain, sunset, and waterfall for negatives, and had around 30 fields, 6 of them negative and the rest positive. Each negative test set had 150 bags of each of flower, mountain, sunset, and waterfall. Also, each test set had 120 fields, around 50 serving as positives and the remainder as negatives.

The top two rows of Table 1 summarize the prediction error of our algorithm ("$k_\wedge$"), GMIL-1, and GMIL-2. For comparison purposes, we also give results for the algorithms Diverse Density (Maron & Lozano-Pérez, 1998) and EMDD (Zhang & Goldman, 2001) that operate in the conventional MIL model. The sunset task fits well into the conventional MIL model; hence there is little difference in performance between any of the algorithms on this data set. But since the conjunctive task requires identifying images that have a field and have no sky, we see that the generalized model is required[3]. We also note that the use of an SVM improved prediction accuracy when compared to all algorithms, including GMIL-1 and GMIL-2, which use the same hypothesis space as our algorithm.

6.2. Identifying Trx-fold Proteins

The low conservation of primary sequence in protein superfamilies such as Thioredoxin-fold (Trx-fold) makes conventional modeling methods difficult to use. Wang et al. (2004) propose using multiple-instance learning as a tool for identification of new Trx-fold proteins. They mapped each protein's primary sequence to a bag in the following way. First, they found in each sequence the primary sequence motif (typically CxxC) that is known to exist in all Trx-fold proteins. They then extracted a window of size 214 around it (30 residues upstream, 180 downstream) and aligned these windows around the motif. They then mapped all sequences to 8-dimensional profiles based on the numeric properties of Kim et al. (2000) and used them as inputs to the multiple-instance learning algorithm.

Wang et al. (2004) used GMIL-2 to perform cross-validation tests: 20-fold CV on 20 postivies and 8-fold CV on 160 negatives. So in each round, they trained GMIL-2 on 19 positive proteins plus one of 8 sets of negative proteins, and tested on the held-out positive protein plus the remaining 7 sets of negative proteins. They repeated this for each of the 8 sets of negative proteins. To compare with their results, we performed the same tests with our kernel, EMDD and DD (Table 1).

6.3. Multi-Site Drug Binding Affinity

Dietterich et al. (1997) introduced the conventional MIL model motivated by predicting whether a conformation of a particular molecule would bind to a single site in another molecule. They also described an open problem of how to predict drugs that bind at multiple sites in a single molecule by fitting in several of them simultaneously.

Scott et al. used a generalization of the synthetic data of Dooly et al. (2002) to reflect the notion of drugs where a molecule must bind at multiple sites to be labeled positive. Dooly et al. created their data by first generating a single "artificial receptor" t. Then "artificial molecules" (denoted B_i) were created, each with 3–5 instances per bag. The label of bag B_i was determined as follows. For each $B_{ij} \in B_i$, they computed $E_{B_{ij}}$, which is the binding energy of B_{ij} to t. They then identified the instance $B_{ij} \in B_i$ that most strongly binds to t and set $E_{B_i} = E_{B_{ij}}$. They then normalized E_{B_i} to $[0, 1]$ and thresholded it at $1/2$ to get a binary label as to whether the molecule binds at t. In the generalization used by Scott et al., there are multiple target points ("subtargets"), each of which must bind to some instance in a bag for the bag to be positive. I.e. bag B_i's label is positive iff each subtarget induces a normalized binding energy of at least $1/2$ in some point $B_{ij} \in B_i$.

Scott et al. used Dooly et al.'s modified data generator to build ten 5-dimensional data sets (200 training bags and 200 testing bags), each with 4 subtargets. To test how well our kernel handles higher-dimensional data, we also generated data with dimension 10 and 20, each with 5 subtargets. As with the 5-dimensional data, ten sets were generated, each with 200 training bags and 200 testing bags. Results are at the bottom of Table 1.

[3] Scott et al. (2003) showed that repulsion points are required for this learning task.

6.4. Musk Data Sets

Finally, we tested on the Musk data sets from the UCI repository (Blake et al., 2004), which represent different conformations of various molecules, labeled according to whether they exhibit a "musk-like" odor when smelled by a human expert. Most results reported by others on this data set are based on 10-fold cross-validation on the 92 bags. We performed 10-fold cross-validation experiments on the same 10 partitions used by Dietterich et al. (1997).

For the Musk experiments, the ratio of diagonal entries in the kernel matrix to the off-diagonal entries was often around 10^{50}. So we applied the method of Schölkopf et al. (2002) to solve this problem. We used the sub-polynomial function $x^{1/50}$ to reduce the range of each entry in the Gram matrices and then let SVM^{light} work with the empirical kernels described in Section 5.3.

Table 2 summarizes our results and those from Andrews et al. (2002) with mi-SVM and MI-SVM and their results with EMDD[4]. Results for DD come from Maron and Lozano-Pérez (1998), and "IAPR" is the iterative axis-parallel rectangle algorithm from Dietterich et al.

There are significant improvements when our empirical kernels are used. The empirical kernel with transduction has the second lowest error rate on Musk1 and the lowest error rate on Musk2. Its performance is competitive with IAPR, which is specially tuned for the Musk data sets. Without transduction, our kernel has slightly higher error rates, but it still has better performance compared to algorithms from the conventional MIL model (except for IAPR and DD on Musk 1).

Since our algorithm's time complexity is linear in d and quadratic in n, we expect our speed to be most competitive when d is large and n is moderate. This is what we found when comparing our algorithm (written in C++) to the Java implementation of EMDD by Zhang and Goldman. For the 10-dimensional binding affinity data, EMDD was 60% faster than our algorithm[5], but only 44% faster on the 20-dimensional data. For Musk 2, EMDD was 24% slower than our algorithm, and for Musk 1 it was 675% slower[6]. This occurred despite the fact that we computed the entire Gram matrix a priori rather than simply computing the entries as needed during SVM optimization. Further, since each learning task in applications such as CBIR and drug binding can be treated as a database query (the data set stays fixed but each task involves a different labeling of the training set), one could build the kernel matrix once for the database and reuse it for each query. This would amortize the cost of building the matrix. This is what we did in our Musk experiments, so the total time spent by our algorithm for all 10 folds was 2 hours for Musk 1 on a 750 MHz Sun Blade (compared to 135 hours for EMDD) and 40 hours for Musk 2 (compared to 485 hours for EMDD).

Table 2. Classification error on the Musk data sets. EMDD, mi-SVM, and MI-SVM are from Andrews et al. (2002), DD is from Maron and Lozano-Pérez (1998), and IAPR is from Dietterich et al. (1997).

Algorithms	Musk 1	Musk 2
$k_\wedge$	0.176	0.227
$k_{\wedge\ emp}$ non-transduction	0.120	0.118
$k_{\wedge\ emp}$ transduction	**0.088**	**0.097**
EMDD	0.152	0.151
DD	0.120	0.160
mi-SVM	0.126	0.164
MI-SVM	0.221	0.157
IAPR	**0.076**	**0.108**

7. Conclusions

The conventional MIL model has proven to be a very powerful one with many applications and efficient algorithms. Algorithms GMIL-1 (Scott et al., 2003) and its faster variant GMIL-2 (Tao & Scott, 2004) in Scott et al.'s generalization of the conventional MIL model have enjoyed success in applications that cannot be represented in the conventional MIL model. However, both algorithms are inherently inefficient, preventing scaling to higher-dimensional data. We formulated their algorithms as a kernel that can be used with a support vector machine to learn concepts in their model. We showed that such a kernel is hard to compute in general, and then we presented an FPRAS for it. Finally, we evaluated our kernel empirically.

Our results not only showed competitiveness with other algorithms in terms of generalization error, but also in speed (versus EMDD), especially for data of more than 20 dimensions. We also note that it is trivial to parallelize the computation of our kernel to get an almost linear speedup. Further, since each learning task in applications such as CBIR and drug binding affinity can be treated as a database query, one could build the kernel matrix once for the entire database and reuse it for each query. This would amortize the cost of building the matrix over many queries.

[4] Andrews et al. (2002) point out that the EMDD experiments of Zhang and Goldman (2001) were optimistically biased since they used the test set to choose the final hypotheses. Thus Andrews et al. reran those experiments. We also reran EMDD on the same 10-fold partitioning used by us and Dietterich et al. Those rates are worse than those in Table 2: 0.152 for Musk 1 and 0.206 for Musk 2, though we did not tune EMDD to minimize error.

[5] Training with SVM^{light} took less than ten seconds once the kernel matrix was computed, while kernel computation took on the order of minutes.

[6] Assuming that a Java implementation is at most 10 times slower than a C++ implementation, our algorithm is thus arguably competitive with EMDD on Musk 1.

An open question that remains is: there are many results bounding the generalization error of SVMs, but they assume that the kernel is exactly computed. What effect does an ϵ-approximate kernel (assuming it is a kernel) have on generalization error?

Acknowledgments

The authors thank Tom Dietterich for his Musk partitionings, Qi Zhang, Sally Goldman, and James Wang for the CBIR data (indirectly from James Wang, Corel, and webshots.com), Qi Zhang for his EMDD/DD code and binding affinity data generator, and Thomas Shores and Steven Dunbar for their discussions. They also thank the ICML reviewers for their feedback. This research was funded in part by NSF grants CCR-0092761 and EPS-0091900, and a grant from the NU Foundation. It was also supported in part by NIH Grant Number RR-P20 RR17675 from the IDeA program of the National Center for Research Resources. This work was completed in part utilizing the Research Computing Facility of the University of Nebraska.

References

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2002). Support vector machines for multiple-instance learning. *Advances in Neural Information Processing Systems 15*.

Blake, C., Keogh, E., & Merz, C. J. (2004). UCI repository of machine learning databases. `www.ics.uci.edu/~mlearn/MLRepository.html`.

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

Dooly, D. R., Zhang, Q., Goldman, S. A., & Amar, R. A. (2002). Multiple-instance learning of real-valued data. *Journal of Machine Learning Research*, *3*, 651–678.

Goldman, S. A., Kwek, S. K., & Scott, S. D. (2001). Agnostic learning of geometric patterns. *Journal of Computer and System Sciences*, *6*, 123–151.

Joachims, T. (1999). Making large-scale SVM learning practical. In B. Schölkopf, C. Burges and A. Smola (Eds.), *Advances in kernel methods: Support vector learning*, chapter 11, 169–184. MIT Press.

Karp, R., Luby, M., & Madras, N. (1989). Monte-Carlo approximation algorithms for enumeration problems. *Journal of Algorithms*, *10*, 429–448.

Kim, J., Moriyama, E. N., Warr, C. G., Clyne, P. J. & Carlson, J. R. (2000). Identification of novel multi-transmembrane proteins from genomic databases using quasi-periodic structural properties. *Bioinf*, *16*, 767–775.

Littlestone, N. (1991). Redundant noisy attributes, attribute errors, and linear threshold learning using Winnow. *Proceedings of the Fourth Annual Workshop on Computational Learning Theory* (pp. 147–156). San Mateo, CA: Morgan Kaufmann.

Maron, O., & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. *Advances in Neural Information Processing Systems 10*.

Maron, O., & Ratan, A. L. (1998). Multiple-instance learning for natural scene classification. *Proc. 15th International Conf. on Machine Learning* (pp. 341–349). Morgan Kaufmann, San Francisco, CA.

Ray, S., & Page, D. (2001). Multiple instance regression. *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 425–432).

Schölkopf, B., Weston, J., Eskin, E., Leslie, C., & Noble, W. S. (2002). A kernel approach for learning from almost orthogonal patterns. *Proceedings of the 13th European Conference on Machine Learning* (pp. 511–528).

Scott, S. D., Zhang, J., & Brown, J. (2003). *On generalized multiple-instance learning* (Tech Rept UNL-CSE-2003-5). Dept. of Comp. Sci., Univ. of Nebraska.

Tao, Q., & Scott, S. D. (2004). A faster algorithm for generalized multiple-instance learning. *Proceedings of the Seventeenth Annual FLAIRS Conference* (to appear).

Valiant, L. G. (1979). The complexity of enumeration and reliability problems. *SIAM J. of Computing*, *8*, 410–421.

Wang, C., Scott, S. D, Zhang, J., Tao, Q., Fomenko, D., & Gladyshev, V. (2004) *A study in modeling low-conservation protein superfamilies* (Tech Rept UNL-CSE-2004-0003). Dept. of Comp. Sci., Univ. of Nebr.

Wang, J., & Zucker, J. D. (2000). Solving the multiple-instance problem: A lazy learning approach. *Proc. 17th Int. Conf. on Machine Learning* (pp. 1119–1125).

Weidmann, N., Frank, E., & Pfahringer, B. (2003). A two-level learning method for generalized multi-instance problems. *Proceedings of the European Conference on Machine Learning* (pp. 468–479).

Zhang, Q., & Goldman, S. A. (2001). EM-DD: An improved multiple-instance learning technique. *Neural Information Processing Systems 14* (pp. 1073–1080).

Zhang, Q., Goldman, S. A., Yu, W., & Fritts, J. E. (2002). Content-based image retrieval using multiple-instance learning. *Proc. 19th International Conf. on Machine Learning* (pp. 682–689).