2003

On Generalized Multiple-Instance Learning

Stephen Scott
University of Nebraska - Lincoln, sscott2@unl.edu

Jun Zhang
University of Nebraska - Lincoln, jzhang8@unl.edu

Joshua Brown
University of Nebraska - Lincoln

University of Nebraska-Lincoln, Computer Science and Engineering
Technical Report # TR-UNL-CSE-2003-0005

On Generalized Multiple-Instance Learning

Stephen Scott (sscott@cse.unl.edu) and Jun Zhang (juzhang@cse.unl.edu)
Department of Computer Science, 115 Ferguson Hall, University of Nebraska, Lincoln, NE 68588-0115, USA

Joshua Brown (jbrown@math.unl.edu)
Department of Mathematics, 810 Oldfather Hall, University of Nebraska, Lincoln, NE 68588-0323, USA

Abstract. We describe a generalization of the multiple-instance learning model in which a bag's label is not based on a single instance's proximity to a single target point. Rather, a bag is positive if and only if it contains a *collection* of instances, each near one of a *set* of target points. We list potential applications of this model (robot vision, content-based image retrieval, protein sequence identification, and drug discovery) and describe target concepts for these applications that cannot be represented in the conventional multiple-instance learning model. We then adapt a learning-theoretic algorithm for learning in this model and present empirical results.

Keywords: multiple-instance learning, Hausdorff metric, Winnow, content-based image retrieval, drug discovery, protein sequence identification

1. Introduction

In the conventional multiple-instance learning (MIL) model, a bag's (boolean or real) label is entirely determined by a single instance in the bag, e.g. for boolean labels, the bag's label is a disjunction of the instances' boolean labels, each of which is typically determined by the instance's proximity to a single target point. But this is not sufficient for some problems. In our generalization of this model, the target concept is a *set* of points $C = \{c_1, \ldots, c_k\}$, and the label for a bag $B = \{b_1, \ldots, b_n\}$ is positive if and only if there is a subset of r target points $C' = \{c_{i_1}, \ldots, c_{i_r}\} \subseteq C$ such that each $c_{i_j} \in C'$ is near some point in B. Here r is a threshold indicating the minimum number of target points that must each be "hit" by some point from B (the same point in B could hit multiple target points). In other words, if we define a boolean attribute a_i for each target point c_i that is 1 if there exists a point $b_j \in B$ near it and 0 otherwise, then the bag's label is some r-of-k threshold function over the attributes (so there are k relevant attributes and B's label is 1 iff at least r of these attributes are 1). Note that if $r = 1$ 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. If $r = k$ then a conjunctive model exists, where each target point must be hit by some instance in B.

Our learning model can be extended in an interesting way. If we let $\bar{C} = \{\bar{c}_1, \ldots, \bar{c}_{k'}\}$ be a set of "repulsion" points, then we can require a positive bag to *not* have any points near each point of $\bar{C}' = \{\bar{c}'_{i_1}, \ldots, \bar{c}'_{i_{r'}}\} \subseteq \bar{C}$ in addition to having a point near each point in C'. This means that to be positive, certain feature values have to be present in some points of bag B, but also certain feature values must be *absent*. This will prove useful in some of our experiments, especially those in Section 6.2.2.

Variants of our model have been studied theoretically and experimentally under the name "geometric patterns". We make explicit the connections between this class and our new model, and extend the class of geometric patterns to handle the more realistic requirements of pattern recognition systems (hence the need for our new model). We also examine a variation of one algorithm for learning geometric patterns (Goldman et al., 2001), improve its efficiency, and present the first empirical results of this algorithm. Despite our improvements, Goldman et al.'s algorithm is inherently inefficient (exponential in the number of features). As such, this paper is a proof-of-concept of the efficacy of such a model: we present positive experimental results for low-dimensional data, which motivate us to investigate new algorithms for higher-dimensional generalized MIL problems.

The rest of this paper is organized as follows. Section 2 describes the learning model and Section 3 discusses related work on geometric patterns and conventional MIL. We summarize our extensions to Goldman et al.'s algorithm in Section 4. We discuss application areas of our model in Section 5 (landmark matching for robot navigation, discovery of antagonist drugs, content-based image retrieval, and protein superfamily identification) and evaluate our algorithm on these areas in Section 6. In Section 7 we conclude.

2. The Learning Model

In the concept class of d-dimensional geometric patterns (e.g. Goldman et al., 2001), each *bag* (multiple-instance example) is a multiset of points in d-dimensional space. Loosely speaking, each concept in the class can be thought of as a set of bags that visually resemble each other. The notion of resemblance between two bags P and Q is formalized by the *Hausdorff metric* (Huttenlocher et al., 1993), which is used in computer vision applications. The Hausdorff distance between bags P and Q is

$$\max\left\{\max_{\vec{p}\in P}\left\{\min_{\vec{q}\in Q}\{dist(\vec{p},\vec{q})\}\right\}, \max_{\vec{q}\in Q}\left\{\min_{\vec{p}\in P}\{dist(\vec{p},\vec{q})\}\right\}\right\}, \qquad (1)$$

where $dist(\vec{p},\vec{q})$ is the distance under some norm between points $\vec{p}$ and $\vec{q}$. In other words, if each point in P reports the distance to its nearest neighbor in Q and each point in Q reports the distance to its nearest neighbor in P, then

the Hausdorff distance is the maximum of these distances. A concept is the set of all bags within some distance γ under the Hausdorff metric of some "ideal" bag[1] of $\leq k$ points (Figure 1 gives a one-dimensional example with $\gamma = 1$ and $k = 3$). We then generalize Equation (1) by allowing $dist$ to be a weighted norm, and the weights can vary for each point in the model. By letting $dist$ be the weighted infinity norm, a target concept C is a set of $\leq k$ axis-parallel boxes and a bag B is positive if and only if (1) every point of B lies in some box of C, and (2) every box of C contains a point from B.

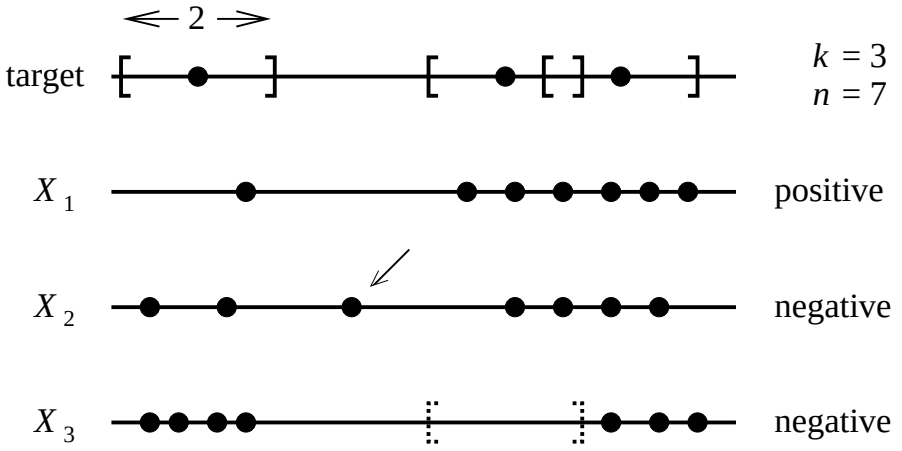

Figure 1. An example target concept of size $k = 3$ points, with three bags X_1, X_2, and X_3.

Intuitively, the above concept class captures the notion of visual similarity quite effectively. However, in pattern recognition applications, this class is not robust against noise or occlusions. Instead, often what is employed is what can be termed *ranked half-Hausdorff*. First, the distance from the model to the pattern is computed, but not vice-versa, since it is assumed that the model is accurate, but that the pattern may include points unrelated to the model. Second, rather than taking the max over the distances from the model to the pattern, instead the sth max is taken, to improve noise tolerance. The final result can be stated as (Huttenlocher et al., 1993; Huttenlocher and Rucklidge, 1992; Sim et al., 1999):

$$\max_{\vec{q} \in Q}^{s} \left\{ \min_{\vec{p} \in P} \{ dist(\vec{p}, \vec{q}) \} \right\} \quad , \tag{2}$$

where $\max^s$ denotes the sth max, P represents the pattern, and Q is the model (i.e. the "ideal" set of points representing the target concept). Again, if Equation (2) uses a weighted infinity norm with variable weights for each

[1] In pattern recognition parlance, the ideal bag is the "model" to which we compare the "images".

point in Q, a target concept C is a set of at most k axis-parallel boxes and a bag B is positive if and only if strictly fewer than s boxes of C do not contain a point from B. This is equivalent to our generalized MIL model with $r = k - s$ (sans repulsion points). Adding a set $\bar{Q}$ of repulsion points is easy as well. In addition to checking if Equation (2) evaluates to at most a constant γ, we check if the following equation evaluates to at least another constant γ':

$$\min_{\vec{q} \in \bar{Q}}^{s'} \left\{ \min_{\vec{p} \in P} \{ dist(\vec{p}, \vec{q}) \} \right\} \quad . \tag{3}$$

So now in addition to requiring that a bag B misses fewer than s boxes from C, we also require that B hits at most s' boxes from $\bar{C}$. In Section 6.2.2 we will show the usefulness of this representation.

3. Related Work

3.1. Learning Geometric Patterns

Some results on learning geometric patterns include when bags come from the real line (Goldberg and Goldman, 1994; Goldberg et al., 1996; Goldman and Scott, 1999), and when they come from a discrete d-dimensional space and have binary labels (Goldman et al., 2001) or real-valued labels (Goldman and Scott, in press). Scott (2000) gives an overview of these algorithms.

We now summarize the algorithm of Goldman et al. (2001) to learn d-dimensional patterns, which we adapt, implement, and evaluate in our new model in Section 6. First note that a discretized space is required for their algorithm: for simplicity, assume that the feature space is $\mathcal{X} = \{1, \ldots, s\}^d$. Their algorithm enumerates all the possible $N = (s(s+1)/2)^d$ boxes in the space and creates two attributes for each box b: a_b and $\bar{a}_b$. Given a bag $B \in \mathcal{X}^n$, the algorithm sets $a_b = 1$ if some point from B lies in b and $a_b = 0$ otherwise. Then[2] $\bar{a}_b = 1 - a_b$. These $2N$ attributes are then given to Winnow[3] (Littlestone, 1988), which learns a linear threshold unit.

Using the above remapping of bags to boolean attributes, the complement of a target concept of geometric patterns can be represented as a monotone disjunction over the attributes, i.e. a disjunction with no negated variables. To see this, recall from Section 2 the two criteria that must be satisfied for a bag B to be positive: (1) each point in B must lie in some box in the target concept, and (2) each box in the target concept must contain a point from

[2] Both these complementary attributes are required since Winnow only learns monotone disjunctions (defined below).

[3] Winnow is very similar to the Perceptron algorithm but updates its weights multiplicatively rather than additively, which often yields much faster convergence, especially in cases like ours when most inputs are irrelevant.

B. Let $C = \{c_1, \ldots, c_k\}$ be the boxes in the target concept and let $C' = \{c'_1, \ldots, c'_{k_{comp}}\}$ be a set of boxes whose union is exactly the complement of the union of the boxes of C. Then Criterion 1 is violated iff some box $c'_i \in C'$ contains a point from B, i.e. if $a_{c'_i} = 1$. Criterion 2 is violated iff some box $c_j \in C$ is empty, i.e. if $\bar{a}_{c_j} = 1$. Thus for a bag B, the following disjunction is true iff B is negative:

$$\left(\bigvee_{c' \in C'} a_{c'} \right) \vee \left(\bigvee_{c \in C} \bar{a}_c \right) \ ,$$

which is monotone since both a and $\bar{a}$ are provided as original inputs to Winnow. Therefore by inverting the bags' labels and applying the remapping, the problem of learning geometric patterns is reduced to learning a monotone disjunction.

Applying a well-known result of Winnow's mistake bounds for learning monotone disjunctions allows us to conclude that Goldman et al.'s mistake bound for this problem is $O((k+k_{comp})d \log s)$. However, its time complexity is $\Omega(s^{2d})$, which is exponential in both $\log s$ and d (though they assumed d was constant). As an example of why this is a problem even for small d, consider the case where each point can only take on 20 values in each of 4 dimensions. Then the number of attributes to Winnow in this case is $2 \cdot 210^4 > 3.8 \times 10^9$. To address this issue, Goldman et al. partitioned the set of N boxes into *groups* such that it is guaranteed that for each box b in a group G, all attributes a_b have the same weight in Winnow, as do all attributes $\bar{a}_b$. Thus their algorithm maintains only one representative box per group G and exactly computes Winnow's weighted sum by summing the products of each group's weight and its size:

$$\sum_{i=1}^{N} a_i \, w_{a_i} + \bar{a}_i \, w_{\bar{a}_i} = \sum_{G \in \mathcal{G}} |G| \, (a_G \, w_{a_G} + \bar{a}_G \, w_{\bar{a}_G}),$$

where $|G|$ is the number of boxes in group G, a_G is the attribute for group G's representative box, and $\mathcal{G}$ is the set of all groups. The set of groups is built by using the points from all bags to rectilinearly partition $\mathcal{X}$ (see Figure 2). The result is a set of *regions* such that any pair of boxes b_i and b_j are in the same group if both have their "lower left" corners in region $R_{\vec{w}}$ and their "upper right" corners in region $R_{\vec{z}}$ (they also defined groups that had both their lower left and upper right corners in the same region). It is easy to see that when the groups are constructed this way, for each pair of boxes b and b' in the same group G, b and b' contain the same subset of points from all bags. Thus $a_b = a_{b'}$ and $\bar{a}_b = \bar{a}_{b'}$ for all bags, and all the Winnow weight updates are the same for attributes a_b and $a_{b'}$ as well as $\bar{a}_b$ and $\bar{a}_{b'}$. Using this construction, at most $O(m^{2d+1})$ groups are built, where m is the number of points used to

partition the space. Thus the exponential dependence on $\log s$ is removed, but the exponential dependence on d remains. (It is unlikely that a polynomial algorithm exists for this learning problem, since this would yield an efficient algorithm for on-line learning of DNF formulas, which is unlikely to exist (Goldman et al., 2001).)

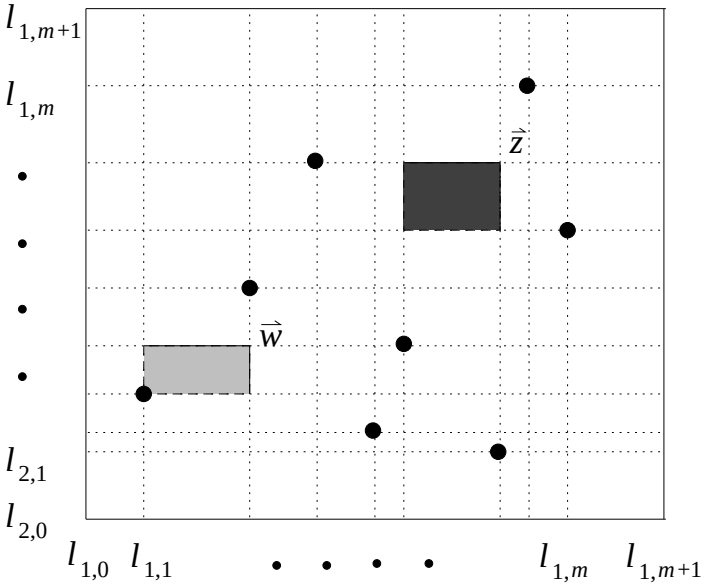

Figure 2. An example of how Goldman et al. construct their groups of boxes.

3.2. Conventional Multiple-Instance Learning

The conventional MIL model was introduced by Dietterich et al. (1997). In their model, each bag is classified as positive if and only if at least one of its elements is labeled as positive by the target concept. Their work was motivated by the problem of predicting whether a molecule would bind at a particular site. They argued empirically that axis-parallel rectangles are good hypotheses for this and other similar learning problems. This MIL model has been extensively studied (Zhang et al., 2002; Long and Tan, 1998; Auer et al., 1997; Auer, 1997; Maron and Lozano-Pérez, 1998; Maron, 1998; Maron and Ratan, 1998; Blum and Kalai, 1998; Zhang and Goldman, 2001; Ruffo, 2000; Wang and Zucker, 2000; Ramon and Raedt, 2000; Andrews et al., 2002), along with extensions for real-valued labels (Dooly et al., 2002; Ray and Page, 2001). In most MIL work, the label of a bag depends only on the label of a single point, and the label of each point typically is assumed to depend on a single target point. Exceptions include some work of Maron and Lozano-Pérez (1998) and Maron and Ratan (1998) in which a target

concept can be a disjunction over multiple points. They also (in a subset of their experiments) mapped each pair of instances to a new instance and added spatial information about the instance pair, which defined a pairwise conjunctive type of learning model. However, they found that allowing more than 2 disjuncts in the target concept or taking more than 2 instances at a time proved computationally very difficult.

In other work, De Raedt (1998) generalizes MIL in the context of inductive logic programming and defines an interesting framework connecting many forms of learning. One of his generalizations allows relations between instances. However, the transformations he gives between the models have exponential time and space complexity.

An issue that arises with the algorithms EMDD (Zhang and Goldman, 2001) and DD (Maron and Lozano-Pérez, 1998) is the need to scale the axes of the feature space so that irrelevant attributes are ignored. Thus the dimensionality of the search space is twice the number of attributes, since each attribute has a scale factor. In contrast, our representation that uses axis-parallel boxes corresponds to a weighted infinity norm, so the axis rescaling is done implicitly. In addition, the rescaling is done independently for each target point, so point c_i could have a completely different rescaling system than point c_j. Also, in Maron et al.'s work with multiple disjuncts or pairwise-combined instances, such generalizations must be explicitly tuned, whereas in our algorithm, the "right" number of boxes is learned.

4. Our Algorithm

The results of Goldman et al. (2001) (Section 3.1) are purely theoretical. When implementing it for application to generalized multiple-instance learning problems, we made certain changes. First, if we only use the attributes $\bar{a}_b$ (which $= 1$ iff box b does not contain a point from the current bag), then we accommodate the half-Hausdorff metric[4] of Equation (2) that generally tolerates more noise (Section 2). (Though we also note that using Full accommodates repulsion points in a target concept, similar to Equation (3) except that it cannot consider the separate values s and s', only $s + s'$. Thus Full will prove useful in some experiments, especially those in Section 6.2.2.) Second, we note that Winnow can learn r-of-k threshold functions rather than simple disjunctions with a factor of r increase in the mistake bound. Thus Half automatically handles ranked half-Hausdorff and Full automatically handles a natural definition of ranked full-Hausdorff. Third, in some experiments we use clustering on the training set to preprocess the data to significantly reduce the number of points (m in Section 3.1) used to build the groups,

[4] We refer to this algorithm as "Half" and the version that uses both a_b and $\bar{a}_b$ as "Full." See Section 6 for a discussion of the relative performances of these algorithms.

hence speeding up the algorithm. Fourth, other heuristic modifications were made to speed up training, employing approximate methods to do bookkeeping in order to reduce training time. Finally, when training is finished, only the attributes with high weight in Winnow ("heavy groups") are presented as the final hypothesis. This yields a more compact, interpretable hypothesis. Such a representation gives immediate information about what portions of the instance space are most relevant to classifying the bags. Our current approach to pruning is to incrementally remove a fraction of the remaining groups each round until the the new (pruned) classifier's false negative error rate increases unacceptably.

While these improvements significantly reduce the time complexity of the algorithm, we are still unable to run it on high-dimensional data like the "Musk" data set in the UCI repository. This is because the number of groups produced is still exponential in d. Thus we are exploring new ways of defining the groups to yield a number of groups that is polynomial in d and exponential[5] in m, as well as methods (Chawla et al., 2003) to *approximate* Winnow's weighted sums when large numbers of inputs are used. These might be useful when working with high-dimensional data.

5. Application Areas

5.1. The Landmark Matching Problem

Consider the problem of recognizing from a visual image of a robot's current location whether or not it is in the vicinity of a known landmark (where a landmark is a location that is visually different from other locations). Such an algorithm is needed as one piece of a complete navigation system where the robot navigates by planning a path between known landmarks, tracking the landmarks as it goes. Because of inaccuracies in effectors and possible errors in its internal map, when the robot believes it is at landmark L, it should check that it is really near L before heading to another landmark. Then adjustments can be made if the robot is not at L by re-homing to L and/or updating its map. By using images taken near the landmark as positive examples and images taken from not near the landmark as negative examples, the goal is to learn a hypothesis that can accurately predict whether a new image came from near or not near the landmark.

For efficiency, one can use one-dimensional visual data (of a 360° view) to do the matching (Hong et al., 1992; Levitt and Lawton, 1990; Pinette, 1993; Suzuki and Arimoto, 1988). We pre-process the images by placing points in

[5] It is unlikely that a group definition exists that is polynomial in both d and m, since this would yield an efficient algorithm for learning DNF formulas, which is unlikely to exist (Goldman et al., 2001).

a 2-dimensional space that correspond to the magnitude and direction of the signal's derivative[6] (Figure 3). An example target concept is in the bottom of Figure 3. Certainly such a target concept cannot be directly represented in the conventional MIL model, since to ensure that a pattern is positive, it must have a point in every box (or at least r of them in the r-of-k case). Note the varying sizes and shapes in the target that are allowed by the weighted infinity norm. Such flexibility can tolerate variability in the reflectivity of objects in the environment: a highly reflective object will have higher day-to-day variations due to changing lighting conditions than a duller surface, so a "tall" target box will better fit the former, and a "short" box will better fit the latter. In addition, flexibility in the target boxes' widths allow tolerance of small (but significant) and translations and rotations of the robot between image acquisitions.

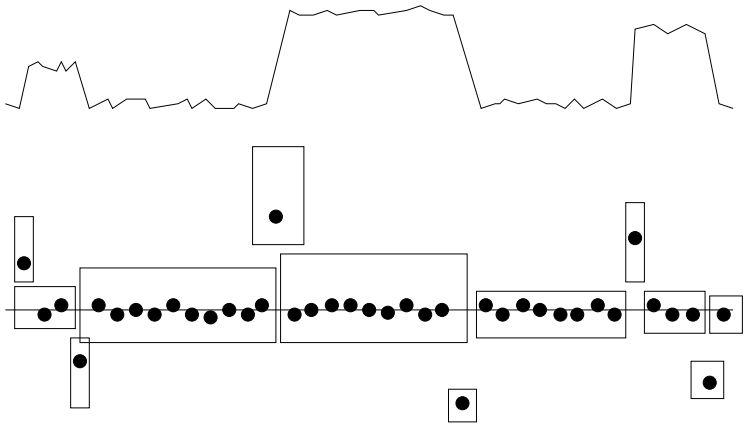

Figure 3. An example of mapping a one-dimensional waveform to a two-dimensional pattern (bag) and an example target concept (the boxes).

Goldman and Scott (1999) presented empirical results on this problem by mapping an input signal to a one-dimensional pattern by placing a point on the line whenever the magnitude of the signal's derivative exceeded a threshold (i.e. by projecting onto the real line all points at the bottom of Figure 3 that are significantly above or below the line). Some of their results were quite good, while others revealed that some crucial information was lost (namely, the direction of change). Thus a two-dimensional pattern can be employed to mitigate this issue. Of course, if the input signal is d dimensions, it can be mapped to a $(d+1)$-dimensional pattern in a similar fashion (Goldman and

[6] In our experiments, we also threshold the derivative, so the points of the bag near the zero line (i.e. those corresponding to little change in the input signal) are filtered out and all that remain are the points representing significant change in the signal.

Scott, in press). This allows one to handle other types of data, e.g. amplitudes of a waveform, sonar data, or temporal difference information.

5.2. Identifying Antagonist Drugs

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. But an open problem is how to predict "antagonist drugs", whose jobs are to bind at multiple sites in a single molecule by fitting in several of them simultaneously. This cannot be represented by a single axis-parallel box in the standard MIL model, but there is a natural representation of this in our model by setting $r = k$. Such a target concept would typically be defined in a very high-dimensional space, since a representation of a conformation is very complex (Dietterich et al., 1997).

5.3. Content-Based Image Retrieval

Maron and Ratan (1998) explored the use of conventional MIL for content-based image retrieval (CBIR) for images of natural scenes. The system is *query by example*, where the user presents examples of desired images, and the system's job is to determine commonalities among the query images. They filtered and subsampled their images and then extracted "blobs" (groups of m adjacent pixels), which they mapped to a $(3m)$-dimensional space (one attribute per RGB pixel value). Each blob was mapped to one point in a bag, and all bags derived from query images were labeled as positive. Then the system used the MIL algorithm diverse density (DD) (Maron and Lozano-Pérez, 1998) to learn a hypothesis, find candidate images in the database, and present those to the user for labeling for more learning. This work was extended by Zhang et al. (2002), who compared DD to EMDD (Zhang and Goldman, 2001) on data sets preprocessed with numerous feature extraction methods, including RGB profiling of blobs (as in Maron and Ratan) and YCrCb (luminance-chrominance) representations coupled with wavelet coefficients (Daubechies, 1988) to represent texture. They also used other (segmentation-based) partitioning methods on the images, and examined multiple ways of selecting the final hypothesis after training.

In general, Zhang and Goldman found few significant differences in prediction performance between DD and EMDD, but they did see significant differences between feature extraction methods and hypothesis selection methods. Most importantly, they observed that in their experiments, it was likely that not one but several target points (in a disjunctive form) were responsible for labeling the examples. In addition, it is arguable that if the target concept were conjunctive (e.g. the desired images contain a field *and* a sky), then the standard MIL model will not work and a more expressive hypothesis is needed. This is especially true if a criterion for being positive included the

absence of certain features, e.g. the images contain a field and contain *no* sky. Such a notion cannot be represented by DD or EMDD, but Full captures this with the a_b attributes of Section 3.1. (Recall that Full's disjunction is true when it assigns a label of negative, so if $a_b = 1$ for a target box b that should be empty, then Full will predict negative.)

In addition, while blobs are useful for representing relative positions of colors (e.g. from Maron and Ratan: a blue blob above a white blob above a brown blob implies a mountain), taken alone they are incapable of representing complex shapes invariant of rotation, translation, and scale. In contrast, features such as line segment descriptors, centralized moments, and elongatedness (Ballard and Brown, 1982) can represent shapes in this way, but only when taken in combination, i.e. using a more complex MIL target concept. Of course, more expressive features may be used on their own to describe shape, such as chain codes or Fourier coefficients of shapes. However, the former requires the dimension of the feature space to vary (since chain codes vary in length for different shapes), so it would be difficult to embed such descriptors into any MIL model. Further, while the latter can be used in a fixed-dimensional feature space, it (along with chain codes) will only represent a single shape (or one of a choice of shapes) if passed through a disjunction. In contrast, a generalized MIL concept can represent the case when combinations of such shapes are required in an image.

As a simple example of using a generalized multiple-instance target concept with shape-based features, imagine representing a human face. 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, we want (for the mouth) one region with elongation near 8 and 0 or 1 holes in the region. Then we have other constraints (in terms of the above features or based on other shape descriptors) for the shape of the face. It is unlikely that such a conjunction of feature constraints can be represented with the conventional disjunctive multiple-instance learning model. In contrast, a target concept defined as an r-of-k threshold function over these features not only can represent a human face, but can also handle occlusion: we could allow both eyes to be represented but the mouth hidden, or we could look for the mouth and shape of the head to allow for eyes to be behind sunglasses.

5.4. PROTEIN SUPERFAMILY IDENTIFICATION

Generally, proteins are added to databases much faster than they can be tested to determine to which *superfamily*[7] they belong. Thus a fundamental problem in computational biology is searching protein databases to find candidate

[7] Protein families generally consist of proteins with similar function, and superfamilies are collections of related families.

```
                          *   *
1A8L:    ...KLIVFVRKDHCQYCDQLKQLVQEL...
1BED:    ...PVVSEFFSFYCPHCNTFEPIIAQL...
1QK8:A   ...LVFFYFSASWCPPCRGFTPQLIEF...
1F9M:A   ...PVVLDMFTQWCGPCKAMAPKYEKL...
1MEK:    ...YLLVEFYAPWCGHCKALAPEYAKA...
```

Figure 4. Alignment of segments of five Trx-fold proteins, indexed by PDB ID.

members of a specific superfamily, which can then be tested in the lab to verify membership. Many successful search methods are based on hidden Markov models (HMMs, e.g. Durbin et al., 1998) built on the amino acid sequences of known members of the superfamily. This method is useful if the superfamily exhibits *primary sequence conservation*, i.e. if the its sequences are similar to each other in their chains of amino acids (represented by letters from a 20-character alphabet). However, some superfamilies of proteins, such as thioredoxin fold proteins (Trx) , lack this property, making HMM-based models difficult to apply. For example, in Figure 4, segments of five Trx-fold proteins are shown with each sequence identified by its ID from the Protein Data Bank (http://www.rcsb.org/pdb/). Only the two cysteines (C, marked by asterisks) are fully conserved in the alignment and very little else is even partially conserved. This is typical of Trx-fold proteins.

Recently, Kim et al. (2000) studied this problem as applied to G Protein-Coupled Receptors. They processed each candidate protein, computing for each amino acid seven properties: GES hydropathy, Kyte-Doolittle index, polarity, pI, α helix index, molecular weight, and solubility. For each property, one value is computed per amino acid, so the sequence of n amino acids is transformed to a sequence of n numbers. Kim et al. then computed summary statistics (mapping to a single-instance learning model) of these numeric sequences and inferred a linear discriminant function to separate GPCRs from non-GPCRs. Their results were good for GPCRs, but Trx has been much more difficult to learn in a single-instance model (Scott et al., 2003), because mapping the sequences to their summary statistics loses significant information that reveals fundamental properties of the proteins (e.g. that in all the Trx-fold proteins, positions k through ℓ have high solubility). Such information is not only useful in identifying new members of the family, but it also may reveal to a biochemist in more detail why certain proteins belong to a family and others do not. This information might also be useful in other applications such as predicting the function of a specific protein by measuring similarities of a candidate protein's signature to the signatures of proteins with known function.

6. Experimental Results

We now describe our experimental results, comparing our algorithm to EMDD (Zhang and Goldman, 2001) and DD (Maron and Lozano-Pérez, 1998). Due to our algorithm's time complexity, we restrict the scope of our experiments to low-dimensional data. Thus our results are meant to argue that using generalized MIL is superior (or at least comparable to) the conventional model for some applications, though a more efficient algorithm than ours is required for higher-dimensional data.

Since EMDD produces several candidate hypotheses, it must select a final hypothesis. In our experiments we let EMDD choose based on performance on an independent validation set. In Sections 6.2 and 6.4, separate validation sets are explicitly built (giving EMDD a *de facto* larger training set). In Sections 6.1 and 6.3, we set aside 20% of the training set for validation.

6.1. The Landmark Matching Problem

Our data for the landmark matching problem comes from Pinette's (1993) "room" data set, which was also used to empirically analyze the algorithm of Goldman and Scott (1999). This set consists of one-dimensional images (*signatures*) taken in a room at coordinates set 1 foot apart (see Figure 5). The signatures (images) taken at these points were light intensities at 1° intervals, so each array was 360 pixels wide. The signatures were grouped together into 4 sets of size 7 each: $\mathcal{K}$, $\mathcal{L}$, $\mathcal{M}$, and $\mathcal{N}$, based on proximity to each other (all points in a single group were within 3 feet of their "landmark", which is the circled point in each group). Since this data set was not collected for our experiments, very little is available: only 7 signatures were collected for each group.

The signatures were thresholded via a spike detector that marked the locations in the image where the absolute value of the change in intensity exceeded 0.09 times the difference between the maximum and minimum. (a threshold of 0.09 is the same that produced the overall best results in the experiments of Goldman and Scott). Then points were placed in the bag at the marked locations, the height of each point equal to the derivative of the signature at that point. In our experiments, we chose one of $\mathcal{K}$, $\mathcal{L}$, $\mathcal{M}$ and $\mathcal{N}$ for the positive bags and two of the remaining sets for the negative bags, yielding 7 positive bags and 14 negative bags in each combination. Leave-one-out cross validation was used.

Results are summarized in Figures 6 and 7 and presented in detail in Table 1. In the figures and the table, "a/bc" corresponds to the data set with positives from set a and negatives from sets b and c. In the table, each false positive (FP, i.e. error rate on the negative bags) and false negative (FN, i.e. error rate on the positive bags) rate is given with one standard deviation and

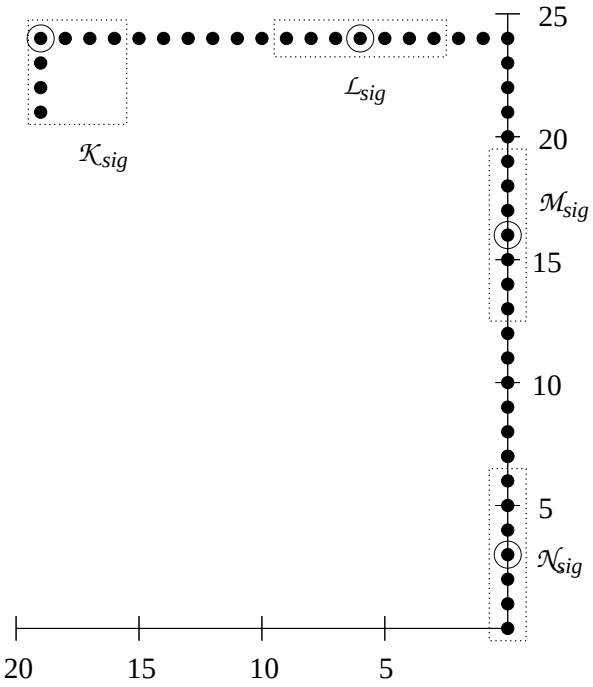

Figure 5. Coordinates where the one-dimensional images (signatures) of the "room" data set were shot. Units on the axes are in feet. Dotted boxes indicate groups of signatures that were given to the algorithm as training and testing data. The circled points highlight the center point of each data set; this point could be considered the target location when that set's patterns were used as positive bags.

emboldened numbers indicate where these intervals do not overlap with those from EMDD[8] (results were similar when comparing intervals to DD). When EMDD had error less than one of our algorithms, the standard deviation intervals always overlapped. For DD, FN error on $\mathcal{L}/\mathcal{KM}$ was less than that for Full, and FP error on $\mathcal{N}/\mathcal{KM}$ was less than that for Half, both with non-overlapping intervals. The results tell us that a hypothesis more expressive than the standard MIL model significantly improves performance for most of the robot data sets. This was especially the case for Full. Further, more often than not, Full converged faster than (or almost as fast as) Half, and performed better on the test examples. This suggests that Full is not overfitting the data (contrary to Section 2), but the small sizes of the data sets makes it difficult to assess the significance of the results.

As an aside, we also note that one motivation for the algorithm of Goldman et al. (2001) (on which our algorithm is based) was that much valuable information is lost when going from a signature to a one-dimensional pattern

[8] However, the small data set size makes assessing statistical significance difficult.

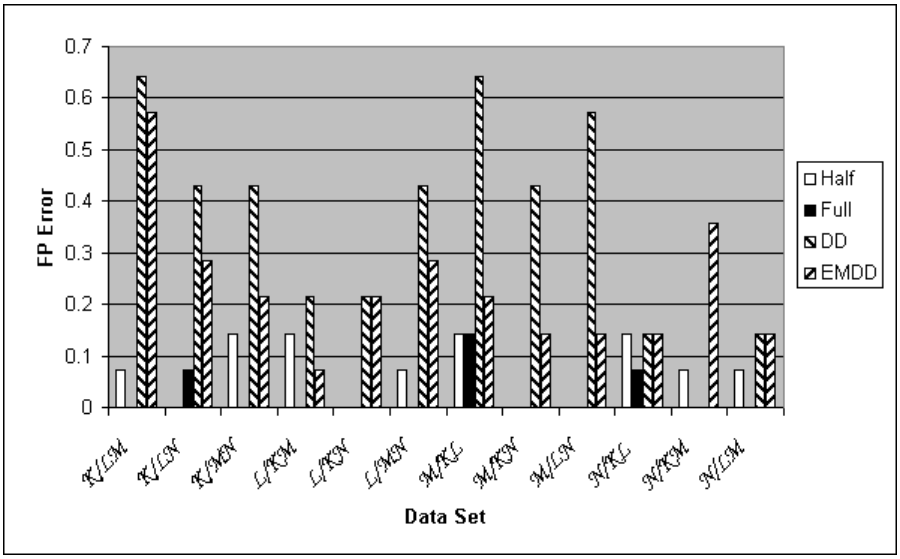

Figure 6. False positive error of each algorithm on the landmark matching task.

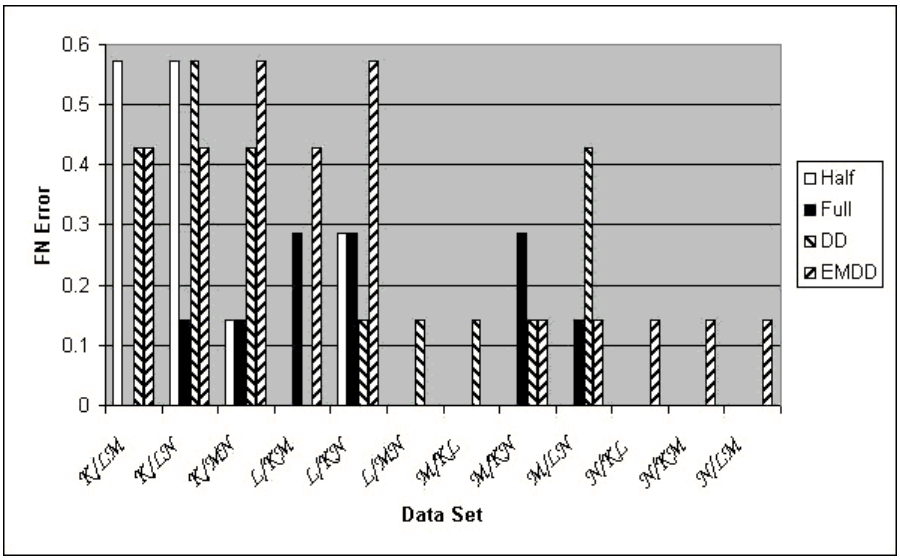

Figure 7. False negative error of each algorithm on the landmark matching task.

Table 1. Detailed results for the landmark matching task.

	Half		Full	
Dataset	FP	FN	FP	FN
$\mathcal{K}/\mathcal{LM}$	**0.0714 ± 0.069**	0.5714 ± 0.187	**0 ± 0**	**0 ± 0**
$\mathcal{K}/\mathcal{LN}$	**0 ± 0**	0.5714 ± 0.187	**0.0714 ± 0.069**	0.1429 ± 0.132
$\mathcal{K}/\mathcal{MN}$	0.1429 ± 0.094	**0.1429 ± 0.132**	**0 ± 0**	**0.1429 ± 0.132**
$\mathcal{L}/\mathcal{KM}$	0.1429 ± 0.094	**0 ± 0**	**0 ± 0**	0.2857 ± 0.171
$\mathcal{L}/\mathcal{KN}$	**0 ± 0**	0.2857 ± 0.171	**0 ± 0**	0.2857 ± 0.171
$\mathcal{L}/\mathcal{MN}$	**0.0713 ± 0.069**	0 ± 0	**0 ± 0**	0 ± 0
$\mathcal{M}/\mathcal{KL}$	0.1429 ± 0.094	0 ± 0	0.1429 ± 0.094	0 ± 0
$\mathcal{M}/\mathcal{KN}$	**0 ± 0**	**0 ± 0**	**0 ± 0**	0.2857 ± 0.171
$\mathcal{M}/\mathcal{LN}$	**0 ± 0**	**0 ± 0**	**0 ± 0**	0.1429 ± 0.132
$\mathcal{N}/\mathcal{KL}$	0.1429 ± 0.094	**0 ± 0**	0.0714 ± 0.069	**0 ± 0**
$\mathcal{N}/\mathcal{KM}$	**0.0714 ± 0.069**	**0 ± 0**	**0 ± 0**	**0 ± 0**
$\mathcal{N}/\mathcal{LM}$	0.0714 ± 0.069	**0 ± 0**	**0 ± 0**	**0 ± 0**

	DD		EMDD	
	FP	FN	FP	FN
$\mathcal{K}/\mathcal{LM}$	0.6429 ± 0.128	0.4286 ± 0.187	0.5714 ± 0.136	0.4286 ± 0.187
$\mathcal{K}/\mathcal{LN}$	0.4286 ± 0.132	0.5714 ± 0.187	0.2857 ± 0.121	0.4286 ± 0.187
$\mathcal{K}/\mathcal{MN}$	0.4286 ± 0.132	0.4286 ± 0.187	0.2143 ± 0.110	0.5714 ± 0.132
$\mathcal{L}/\mathcal{KM}$	0.2143 ± 0.110	*0 ± 0*	0.0714 ± 0.069	0.4285 ± 0.187
$\mathcal{L}/\mathcal{KN}$	0.2143 ± 0.110	0.1429 ± 0.132	0.2143 ± 0.110	0.5714 ± 0.187
$\mathcal{L}/\mathcal{MN}$	0.4286 ± 0.132	0.1429 ± 0.132	0.2857 ± 0.121	0 ± 0
$\mathcal{M}/\mathcal{KL}$	0.6429 ± 0.128	0.1429 ± 0.132	0.2143 ± 0.110	0 ± 0
$\mathcal{M}/\mathcal{KN}$	0.4286 ± 0.132	0.1429 ± 0.132	0.1429 ± 0.094	0.1429 ± 0.132
$\mathcal{M}/\mathcal{LN}$	0.5714 ± 0.132	0.4286 ± 0.187	0.1429 ± 0.094	0.1429 ± 0.132
$\mathcal{N}/\mathcal{KL}$	0.1429 ± 0.094	0 ± 0	0.1429 ± 0.094	0.1429 ± 0.132
$\mathcal{N}/\mathcal{KM}$	*0 ± 0*	0 ± 0	0.3571 ± 0.128	0.1429 ± 0.132
$\mathcal{N}/\mathcal{LM}$	0.1429 ± 0.094	0 ± 0	0.1429 ± 0.094	0.1429 ± 0.132

as done by Goldman and Scott (1999). Some of Goldman and Scott's most difficult data sets to learn with were those that got their positives from set $\mathcal{L}$, and it was believed that this was due in part to this loss of information. Indeed, Goldman and Scott's FN rate on $\mathcal{L}/\mathcal{KM}$ is 0.5714 ± 0.187, their FP rate on $\mathcal{L}/\mathcal{KN}$ is 0.1429 ± 0.094, and on $\mathcal{L}/\mathcal{MN}$ their FP rate is 0.0714 ± 0.069 and their FN rate is 0.4286 ± 0.187. All of these have standard deviation intervals

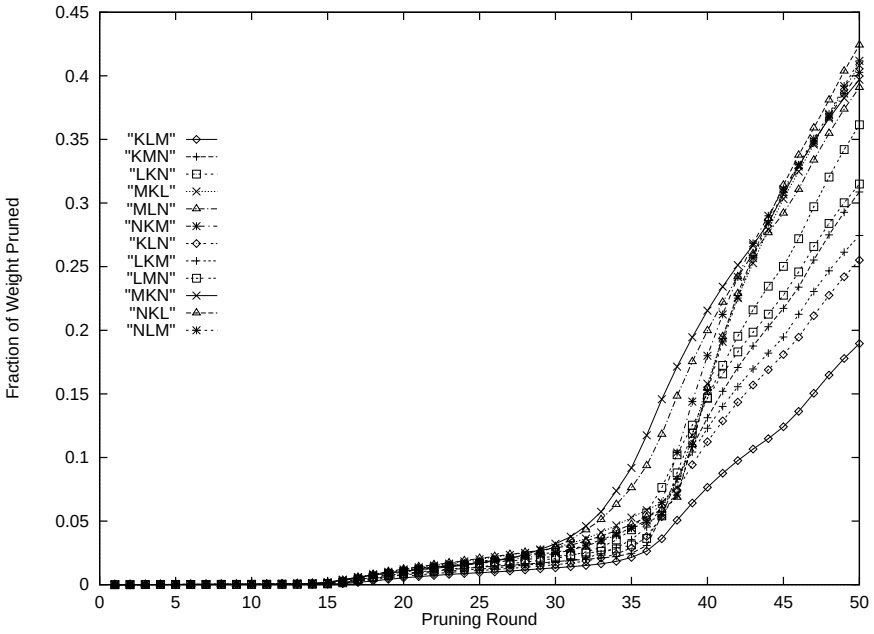

Figure 8. Fraction of weight pruned versus pruning round for Half on the robot data sets.

that do not overlap one or both of Half and Full (most of the other error rates are comparable to ours).

We tested the effect of group pruning by reducing the number of groups by 10% each round for 50 rounds. The results are summarized in Figures 8–11. Figures 8 and 10 show the average cumulative fraction of the weight that is pruned by each round for each of the 12 data sets for Half and Full. We see that before round 35, very little weight was pruned from any hypothesis, despite having pruned $1 - 0.9^{35} = 0.975$ of the groups, implying that relatively few of Winnow's inputs are relevant. After this point, the fraction of weight pruned increased dramatically. Not surprisingly, this corresponds to a significant increase in FP error, as shown in Figures 9 and 11. Therefore we can reduce the number of groups from over 8.96×10^6 to under 2.24×10^5 without significantly altering performance on the training set, but further reductions will adversely affect the hypothesis.

6.2. CONTENT-BASED IMAGE RETRIEVAL

We performed two CBIR experiments[9]. Our first experiment was similar to Zhang et al.'s (2002) "sunset" task: to distinguish images containing sunsets from those not containing sunsets. Our second experiment tests a conjunctive

[9] Based on data from Wang et al. (2001), the Corel Image Suite, and www.webshots.com.

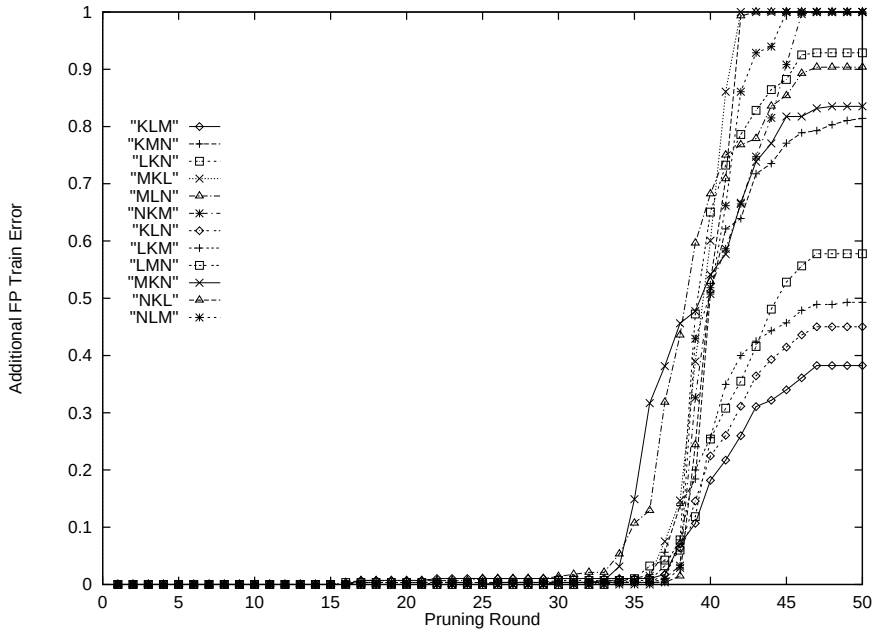

Figure 9. Additional false positive error on the training set versus pruning round for Half on the robot data sets.

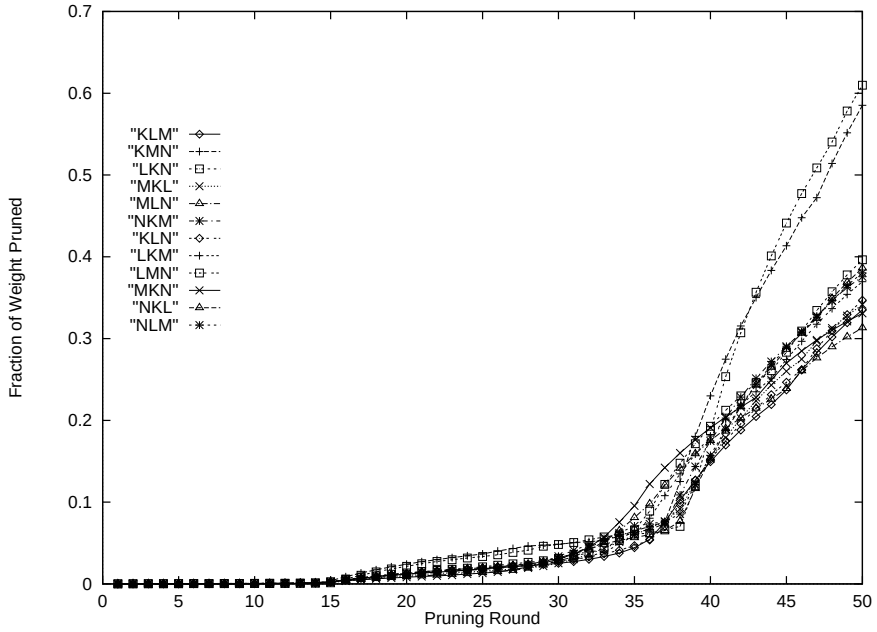

Figure 10. Fraction of weight pruned versus pruning round for Full on the robot data sets.

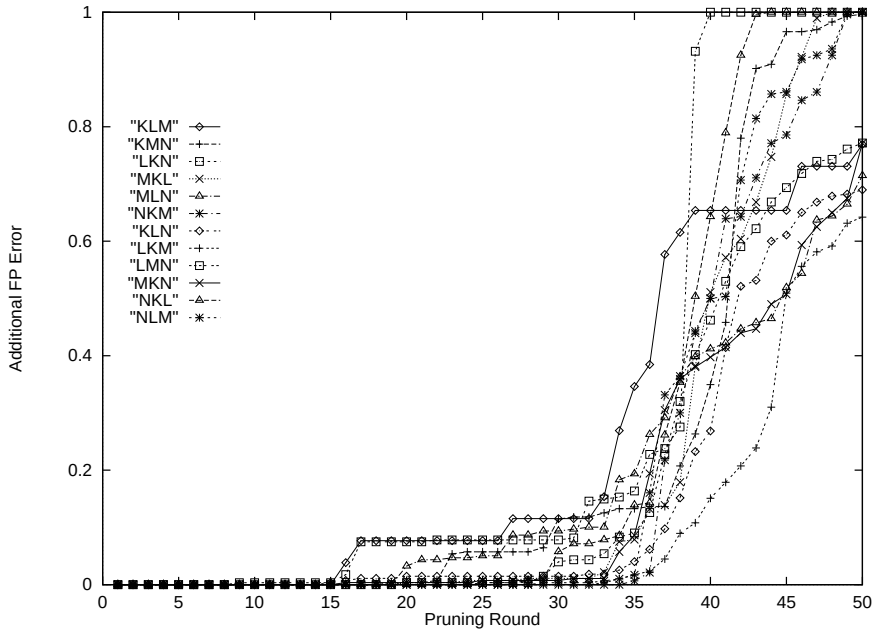

Figure 11. Additional false positive error on the training set versus pruning round for Full on the robot data sets.

CBIR concept, where the goal is to distinguish images containing a field with no sky from those containing a field and sky or containing no field. For these two tasks, we prepared our data the same way, as described below.

Some of Zhang et al.'s best results came from their segmentation-based YCrCb (luminance-chrominance) bag representation with wavelet coefficients. Specifically, they divided each image into 4×4 blobs of pixels, and represented each blob with six features: Y, Cr, Cb, HL(Y), LH(Y), and HH(Y), where the latter three features came from applying Daubechies-4 wavelet transforms (Daubechies, 1988) on the luminance component. They then segmented the image with a k-means segmentation algorithm (Hartigan and Wong, 1979) and for each segment, averaged the 6 features, which relates each segment to a point in the bag that corresponds to the entire image.

Building our set of groups on the entire set of 6-dimensional training bags would generate roughly 10^{50} groups. So to reduce the time complexity, we removed the luminance value from each feature vector, reducing the dimensionality of the feature space to 5. These new bags were used to train and test DD and EMDD. Our algorithm also used these bags, but in defining the groups, we k-means clustered the training data into 6 clusters and used the clusters' point representatives to build the set of groups (so $m = 6$ from Section 4). So the original data was still used, but Winnow could not distin-

guish between any points in the same region of Figure 2. Since EMDD and DD used the original (unclustered) data, they had more information about the data, though the effect of this was unexpected for both the sunset and conjunctive learning tasks (more on this in a moment).

6.2.1. *Sunset Learning Task*

Like Zhang et al., we ran 30 sunset experiments, testing on random sets of 720 examples (120 positives and 600 negatives): 150 negatives each from the waterfall, mountain, field, and flower sets. Training sets consisted of 50 positives and 50 negatives each, and EMDD was given an independent validation set of 40 negatives and 10 positives to choose its final hypothesis. Full[10] had an average FP error of 0.0801 and an average FN error of 0.1567 (it took an average of 42 rounds for Full to converge to 100% accuracy on the training set). DD's FP error on the same data sets was 0.0777 and its FN error is 0.1681. EMDD's FP error was 0.0815 and its FN error is 0.1664. According to a paired t test, there is no statistically significant advantage (even at the 60% level) of Full over either EMDD or DD or vice-versa.

The lack of advantage of Full over EMDD and DD could be explained one of two ways. First, perhaps there is little additional information that can be exploited by generalized MIL. If this is the case, then at least for our choice of features, there is little to be gained from generalized MIL (though perhaps adding back the luminance value or other features would help). Second, perhaps the information lost due to clustering[11] in forming the grid (preventing Winnow from distinguishing between some points) caused additional error for Full.

To help test the latter hypothesis about the effect of clustering, we reran EMDD and DD on data with resolution similar to that used for Full. Specifically, for each feature vector $\vec{x}$ in each of DD's and EMDD's training, testing, and validation bags, we remapped $\vec{x}$ to $\vec{c}_{\vec{x}}$, where $\vec{c}_{\vec{x}}$ is the center of the region (from Figure 2) that $\vec{x}$ lies in when the groups are built based on the $m = 6$ points used by Full. Thus in these new data sets, EMDD and DD cannot distinguish between points in the same region, which is the same constraint that Full operates under. Interestingly, there was no significant change (even at the 60% level) in either algorithms' performance when run on this remapped data, and neither Full nor EMDD/DD had a significant advantage over the other. So for this learning task, the lost information due to clustering did not hurt EMDD or DD. This suggests that increasing m would not help Full, and

[10] As described below, in our sunset experiments, Half had difficulty performing well, and at its best it was still no better than Full. Thus to save time, Half's runs were aborted and only results for Full are presented.

[11] It is also possible that a more refined preprocessing step, e.g. clustering with rescaled dimensions, might squeeze more information out of the data.

instead perhaps there is no other information in the sunset data that can be used by generalized MIL.

Despite the lack of influence of clustering on EMDD and DD, we did notice that Half was influenced by it. In our experiments, Half required at least 1.5–3 times as many training rounds as Full to reach 100% accuracy on the training set, and often even 5 times as many rounds was insufficient. If training was halted prior to reaching 100% accuracy, Half's FN generalization error was ridiculously high. On tests when Half did reach 100% accuracy, its FP and FN generalization error rates were statistically indistinguishable from Full's. In general, experiments in which Half failed to reach 100% training accuracy had the smallest values of m. (Note from Figure 2 that if two or more of the clusters' centers are coincident when projected onto an axis, then we get fewer total regions and thus fewer groups than if all cluster centers are in general position. For our experiments with both Half and Full, m ranged from 1.23×10^6 to 7.26×10^6.) Thus we believe that while EMDD and DD (and probably Full) are not strongly influenced by clustering, Half is. We will evaluate this in more detail in future work.

6.2.2. *Conjunctive Learning Task*

Our second experiment tests a conjunctive CBIR concept, where the goal is to distinguish images containing a field with no sky from those containing a field and sky or containing no field. We relabeled Zhang et al.'s field images from positive to negative those that contained the sky. 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. Validation sets were similarly modified. The dimension and number of clusters was the same as for sunset, and 10 experiments were run.

Full's average FP error was 0.1278 and its FN error was 0.2194. EMDD had 0.2134 FP error and 0.2440 FN error, while DD's FP error was 0.1727 and its FN error was 0.2815. By a paired t test, Full's advantage over EMDD is significant at 95% for FP and at 75% for FN. Full's advantage over DD is significant at 85% for FP and at 95% for FN. This occurred despite the potential loss of information that Full had from clustering. Thus we see that conjunctive CBIR concepts benefit from generalized MIL.

To estimate the effect of clustering on generalization error, we reran EMDD and DD on data remapped as in Section 6.2.1. EMDD's FN error increased to 0.4516 (an increase that is significant at 97.5%) but its FP error decreased to 0.1306 (significant at 95%). DD's FN error increased to 0.4090 (significant at 99%) and its FP error decreased to 0.1313 (significant at 80%). So clustering forced EMDD and DD to focus more on correctly classifying the negative bags (which they were doing already, based on their error rates on the unclus-

tered data), and the increases in FN error far outweigh the decreases in FP error. Thus there is evidence indicating that increasing m would allow Full to perform even better on this learning task. Future work includes rerunning our algorithm with increasing numbers of clusters to measure the effect.

Contrasting Full's performance with Half's, we found that as with the sunset task, Half took longer to reach 100% accuracy on the training set, and without reaching that point, its FN generalization error was very high. However, for this task Half fared much more poorly versus Full than in Section 6.2.1. If Half did reach 100% accuracy, it required around 2.5 times the number of rounds as Full, but unlike Section 6.2.1, when this happened Half still had statistically significantly (at 75%) higher FP and FN error than full. But even more telling is the fact that for some of our data sets, Half failed to reach 100% accuracy even after training for 10–50 times the number of rounds that Full used. While some of this is probably attributable to clustering, a stronger reason for this is the nature of the learning task: field and *not* sky. That is, the target concept will be defined by "field" regions where a point from each positive bag must hit as well as "sky" regions where *no* point from a positive bag may hit (i.e. repulsion points). From Sections 3.1 and 4, we see that Full can represent such a concept but Half cannot.

6.2.3. *Group Pruning*

We tested the effect of group pruning for both learning tasks by reducing the number of groups by 10% each round for 50 rounds. The results are summarized in Figures 12 and 13. Figure 12 shows the average cumulative fraction of the weight that is pruned in each round for Full.

We see that before round 35, very little weight was pruned from any hypothesis for either learning task, despite having pruned $1-0.9^{35} > 0.97$ of the groups, implying that relatively few of Winnow's inputs are relevant. After this point, the fraction of weight pruned increased faster, especially for the conjunctive learning task. Not surprisingly, this corresponds to a significant increase in FP error for the conjunctive task, as shown in Figure 13. Therefore for the conjunctive task, we can reduce the number of groups by almost 98% without significantly altering generalization performance, but further reductions will adversely affect the hypothesis.

Interestingly, for the sunset learning task, even 50 pruning rounds (removing $1 - 0.9^{50} > 0.994$ of the groups) only removed about 11% of the total weight and barely increased false positive generalization error. So certainly fewer attributes are relevant to the sunset concept than to the conjunctive concept, which is reasonable given the nature of the learning tasks.

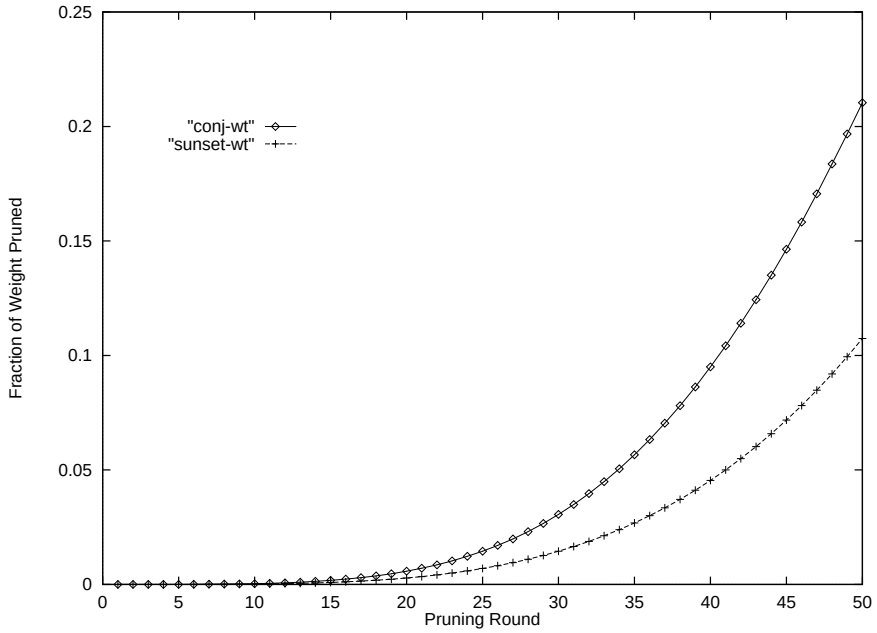

Figure 12. Fraction of weight pruned versus pruning round for Full on the conjunctive and sunset CBIR tasks.

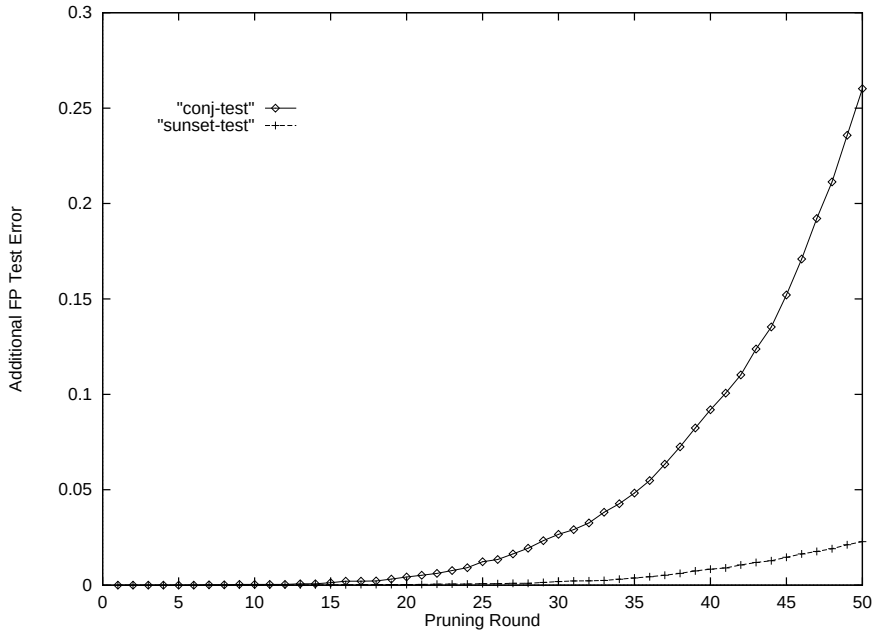

Figure 13. Additional false positive error on the testing sets versus pruning round for Full on the conjunctive and sunset CBIR tasks.

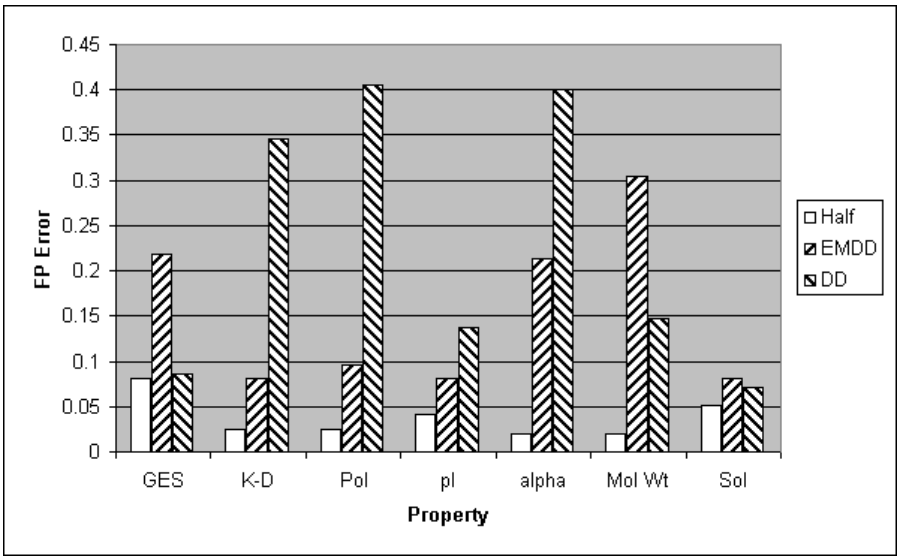

Figure 14. False positive error of each algorithm on the 80% filtered protein data.

6.3. PROTEIN SUPERFAMILY IDENTIFICATION

We ran two tests of our algorithm on protein data. In both tests, we filtered our data to reduce primary sequence similarities, since the goal of this application is to identify new Trx sequences that are highly dissimilar to known ones. First we filtered our data such that no two sequences were more than 80% similar when pairwise aligned, yielding 183 positives and 195 negatives. We then split our data into three sets of approximately equal sizes: A, B, and C. Each of Kim et al.'s (2000) seven properties (listed in Section 5.4) was computed for each amino acid in each sequence in each set. For each property, we ran 3 experiments: training on each pair of sets from $\{A, B, C\}$ and testing on the third. The results[12] are summarized in Figures 14 and 15 and given in detail in Table 2. The "conf" numbers for EMDD's and DD's errors in Table 2 indicate the confidence levels of Half's advantage over them by a paired t test. We see that for these experiments, there is a significant advantage to using a hypothesis from our new model, at least when only one property is used at a time. Further, Half's similar performance to Full indicates that for this experiment, there is no additional information to be gained from where the positive bags do not have points, which is quite different from Section 6.2.2.

[12] After running Full for a subset of these experiments, we saw that those results were very similar to Half's in terms of generalization error and number of training rounds required to reach 100% training accuracy, so we focused on Half.

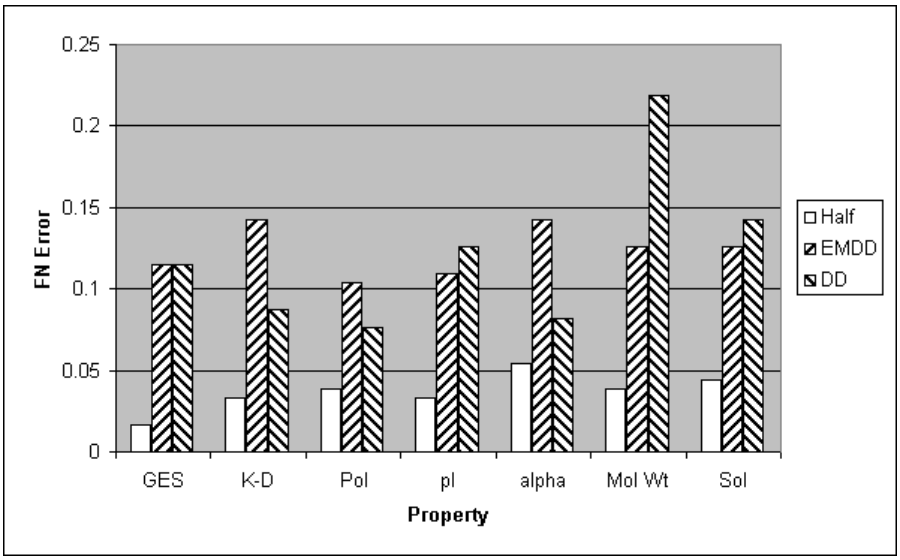

Figure 15. False negative error of each algorithm on the 80% filtered protein data.

Table 2. Results on 80% filtered protein data.

Property	Half		EMDD		DD	
	FP	FN	FP/conf	FN/conf	FP/conf	FN/conf
GES hydropathy	0.0812	0.0164	0.2183/75	0.1148/97.5	0.0863/no	0.1148/97.5
Kyte-Doolittle	0.0254	0.0328	0.0812/90	0.1421/99	0.3452/75	0.0874/75
Polarity	0.0254	0.0383	0.0964/90	0.1038/90	0.4061/75	0.0765/75
pI	0.0406	0.0328	0.0812/75	0.1093/75	0.1371/75	0.1257/95
α helix	0.0203	0.0546	0.2132/75	0.1421/95	0.4010/75	0.0820/90
Mol Wt	0.0203	0.0383	0.3046/90	0.1257/97.5	0.1472/95	0.2186/90
Solubility	0.0507	0.0437	0.0812/75	0.1257/97.5	0.0711/60	0.1421/99

In contrast, we summarize results from Scott et al. (2003), who applied other approaches to this problem. They used the hidden Markov model tool HMMER[13] on the original (primary) sequences (a common approach in biological sequence analysis) and on predicted and true secondary structures[14] of the sequences. Also, similar to Kim et al., they computed summary statis-

[13] http://hmmer.wustl.edu/

[14] When a protein folds on itself in three-dimensional space, amino acids that are near each other together form various higher-order structures, such as α helices and β sheets (Stryer, 1995). A protein's secondary structure is a linear sequence of these higher-order structures.

tics of the properties (e.g. mean and variance of the derivatives) to map the multiple-instance data that we used to single-instance data that they used in C4.5. HMMER on predicted secondary structure had both FP and FN error rates around 0.18, HMMER on true secondary structure had both FP and FN rates around 0.30, and C4.5 on single-instance data had both FP and FN rates around 0.15. Thus our algorithm performed significantly better than these other approaches on these data sets. The only exception is HMMER on the primary sequences, which had FP and FN rates below 0.01. So while generalized multiple-instance learning seems to work significantly better than the conventional MIL model on this data, the sequences are so similar that standard HMM-based approaches perform slightly better. This implies that in fact 80% inter-sequence similarity is high enough for HMMER on primary sequence to be the preferred modeling technique. Since the goal of Scott et al.'s work was to identify new, highly-dissimilar sequences (i.e. ones that cannot be detected by HMMER on primary sequence), we conducted another test.

To further reduce similarities between the sequences in our data set (and more accurately model the problem of finding new Trx-fold protein families), we more aggressively filtered our data. We first hierarchically clustered our sequences with ClustalW[15] and chose the 9 sequences that were least pairwise-similar[16]. Pairwise identity of sequences ranged from 37% to 60%, averaging 47%. The set of negative examples remained the same. We then performed leave-one-out cross-validation on our new set of positives. We also split our set of negatives into 10 sets, trained our algorithms on the 8 positives plus one of the 10 sets of negatives, and tested on the held-out positive plus the remaining 9 sets of negatives. Thus we ran $9 \times 10 = 90$ experiments for each algorithm.

Results[17] are summarized in Figures 16–18 and given in detail in Table 3. FP error is given as the average over all 90 runs and "FN avg" is the average FN rate over all 90 runs. Since each held out positive example was used as a test example in 10 experiments (one for each negative set), we gave an algorithm credit for correctly classifying the held-out positive if it successfully identified it at least half the time. "FN count" is the average over all 9 positives of these counts. Numbers in bold face are FP error rates[18] for Half that are less than those for EMDD and DD with at least 95% confidence according to a paired t test. Of those not in bold, for α helix, Half is lower than DD

[15] http://clustalw.genome.ad.jp/

[16] One reason that the set was so small is that we also had to restrict our set to sequences for which true structural information was available, so we could contrast our results with those of Scott et al. (2003).

[17] Full's FP error was typically much less than Half's but its FN error rates were larger.

[18] We do not assess significance of the FN rates since there are only 9 total positive examples available.

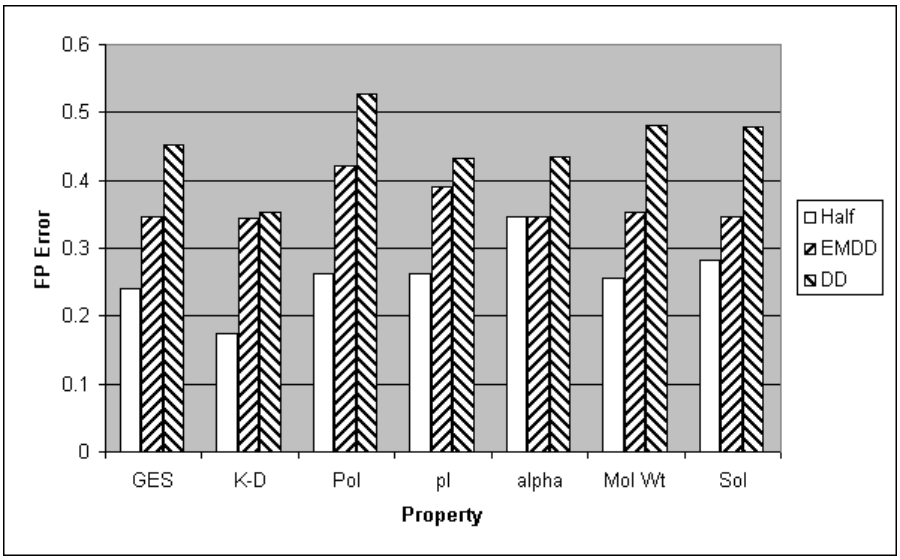

Figure 16. False positive error of each algorithm on the (9×10)-fold cross-validation experiments on protein data.

at 90% confidence but there is no significant difference with EMDD, and for solubility, Half is lower than DD at 99% confidence and lower than EMDD at 75% confidence. Thus our algorithm is much more selective than DD and EMDD, which is logical since each of their hypotheses is represented by a single point. It also explains why their FN errors are lower than ours. However, since it is well-known that all Trx-fold proteins follow simple patterns in their primary sequence (Martin, 1995; Follmann and Haberlein, 1995), it is in fact trivial to get 0 FN error at the expense of high FP error, so a classifier with low FP error would be a useful second-stage filter.

The only results from Scott et al. (2003) that are comparable to ours (i.e. low FP error without FN errors near 1) are their HMMs built on predicted secondary structure[19] (with FP error $= 0.145$ and FN error $= 0.222$), HMMs built on a special alphabet based on hydrophobicity (with FP error $= 0.560$ and FN error $= 0.046$), and C4.5 (with FP error $= 0.333$ and FN error $= 0.329$). So the most well-rounded model for this problem is the HMM built on secondary structure (in contrast to the previous experiment, when the only algorithm competitive with ours was an HMM built on primary sequence). However, our algorithm (as well as DD and EMDD) worked with data that lacked potentially valuable information. First, the MIL algorithms used only one property at a time (done to reduce time complexity). Using all seven prop-

[19] For this experiment, HMMs built on primary sequence had FN error of 0 and FP error of 1.0.

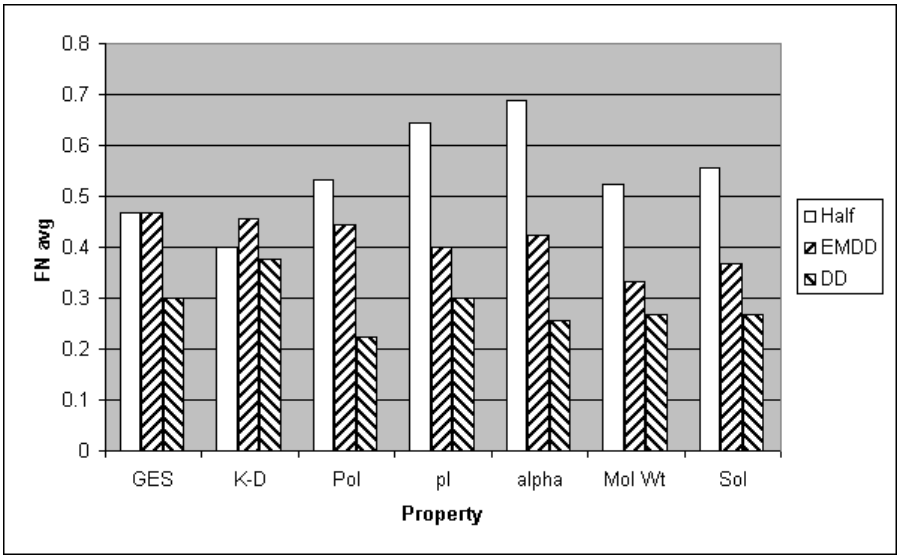

Figure 17. False negative (avg) error of each algorithm on the (9×10)-fold cross-validation experiments on protein data.

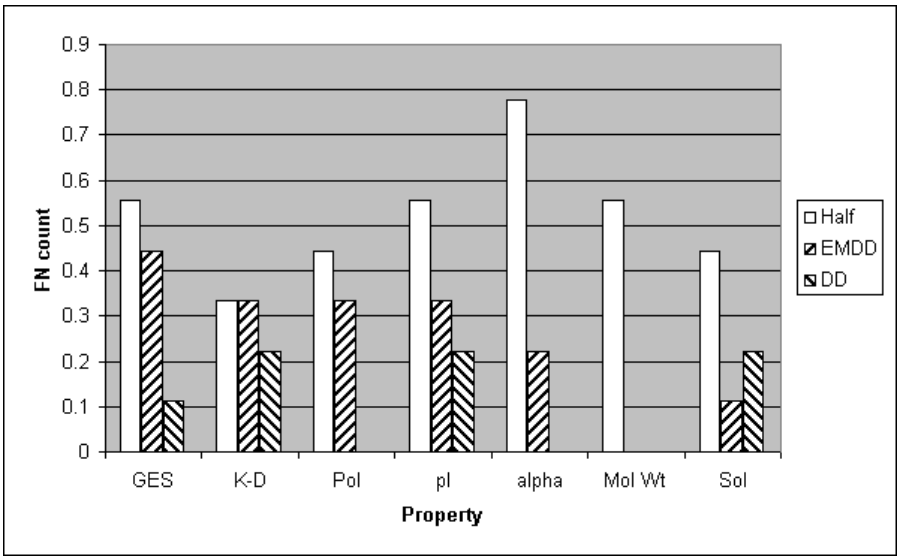

Figure 18. False negative (count) error of each algorithm on the (9×10)-fold cross-validation experiments on protein data.

Table 3. (9×10)-fold cross-validation results on protein data.

Property	Half			EMDD			DD		
	FP	FN avg	FN count	FP	FN avg	FN count	FP	FN avg	FN count
GES hydropathy	**0.2414**	0.4667	0.5556	0.3457	0.4667	0.4444	0.4530	0.3000	0.1111
Kyte-Doolittle	**0.1747**	0.4000	0.3333	0.3440	0.4556	0.3333	0.3530	0.3778	0.2222
Polarity	**0.2617**	0.5333	0.4444	0.4207	0.4444	0.3333	0.5280	0.2222	0.0000
pI	**0.2626**	0.6444	0.5556	0.3895	0.4000	0.3333	0.4314	0.3000	0.2222
α helix	0.3468	0.6889	0.7778	0.3473	0.4222	0.2222	0.4350	0.2556	0.0000
Mol Wt	**0.2549**	0.5247	0.5556	0.3527	0.3333	0.0000	0.4814	0.2667	0.000
Solubility	0.2828	0.5556	0.4444	0.3470	0.3667	0.1111	0.4781	0.2667	0.2222

erties simultaneously could improve performance. Second, after mapping the amino acid sequences to property sequences, we immediately applied our algorithm to the data. In contrast, Kim et al. (2000) (as well as Scott et al. for their C4.5 experiments) smoothed their property sequences with a size-15 Gaussian kernel to filter out noise due to mutations and experimental errors. We did not perform this preprocessing step because applying the kernel would implicitly map the range of the y axis (i.e. the set of possible property values) from a size-20 discrete set (one possible value per amino acid type) to a continuous set. This would blow up our algorithm's time complexity, requiring us to e.g. cluster the data as in Sections 6.2 and 6.4.

Third, when mapping the property sequences to multiple-instance examples, the first coordinate of each point $\vec{b}$ in each bag B relates to the position in the original sequence of $\vec{b}$'s corresponding amino acid. E.g. if amino acid x is 25% downstream from the start of the sequence, then we would expect $\vec{b}_x$'s first coordinate to be about 25% of the way from the origin to the final point. More specifically, since all the MIL algorithms applied to this data set are looking for similar property values in certain regions of each positive bag, it is critical that these regions have similar coordinates in the first dimension, lest the algorithms fail to detect the similarities. Since the lengths of the sequences vary significantly (even among only the positive sequences), correctly aligning them is a challenge. This problem is compounded by the low primary sequence similarity, which prevents us from using conventional multiple sequence alignment algorithms.

To answer the problem of setting the first coordinate, we used a naïve and simple mechanism to align the sequences in our experiments. First we aligned the two conserved cysteines (marked by asterisks in Figure 4) in all

sequences, since this CxxC motif (or a similar[20] one) appears in all Trx-fold proteins. Then (since it is known that all Trx-fold proteins extend at most 180 amino acids beyond the motif) we used the next 180 symbols of each sequence, discarding everything else that lay beyond that point. If a sequence was not long enough to go 180 symbols past the CxxC, it was linearly rescaled so that the last symbol was in position 180. Finally, since it is also known that Trx-fold proteins extend at most 20 positions upstream of the motif, we also used these 20 positions, yielding a sequence of length at most (and often strictly less than) 204, mapped to a space that spans $[1, 204]$. This approach is naïve since it assumes that all of the 180 downstream symbols are in fact part of the Trx fold, which might not be the case. Thus while our results (especially those in Table 2) indicate that our alignment procedure works fairly well, it is possible that the data could be aligned much better.

Ongoing work includes addressing all three of these issues by improving our alignment procedure (possibly by aligning based on the property values themselves) while running a faster version of our algorithm on smoothed, 8-dimensional data.

We tested group pruning by reducing the number of groups by 10% each round for 50 rounds. The results are summarized in Figures 19 and 20. Figure 19 shows the average cumulative fraction of the weight that is pruned by each experiment for each of the 7 properties for Half. In contrast to the group pruning results of Section 6.1, very early on significant weight is pruned (except for Molecular Weight and Solubility), indicating that many more of the Winnow attributes (boxes) are relevant than for our other application areas. As expected, the additional FP error on the test[21] set (Figure 20) also increases earlier than in Section 6.1, though the increase starts at a later pruning round than is suggested by Figure 19. Despite this, we can still reduce the number of groups by about 80–88% (about 15–20 rounds of pruning) without incurring much extra FP error. This corresponds to reducing the number of groups from 3.1×10^6–4.3×10^6 to 4.3×10^5–8.9×10^5.

6.4. Binding Affinity/Antagonist Drugs

We used a generalization of the synthetic data of Dooly et al. (2002) to reflect the notion of "antagonist" drugs (Dietterich et al., 1997) 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. Let B_{ij} denote the jth instance of B_i and B_{ijk} denote the value of feature k in B_{ij}. If t_k is the value of feature k of t and $s_k \in [0, 1]$ is a scale factor indicating the relevance of feature k in binding

[20] Sometimes a serine (S) replaces one of the cysteines, but the motif still exists.
[21] Plots for additional training set error were very similar.

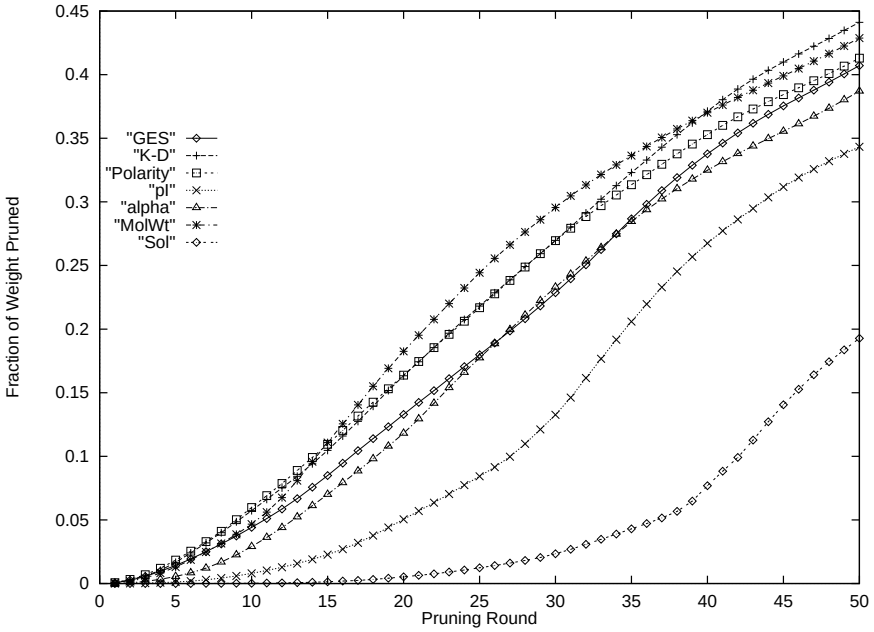

Figure 19. Fraction of weight pruned versus pruning round for Half on the protein data sets.

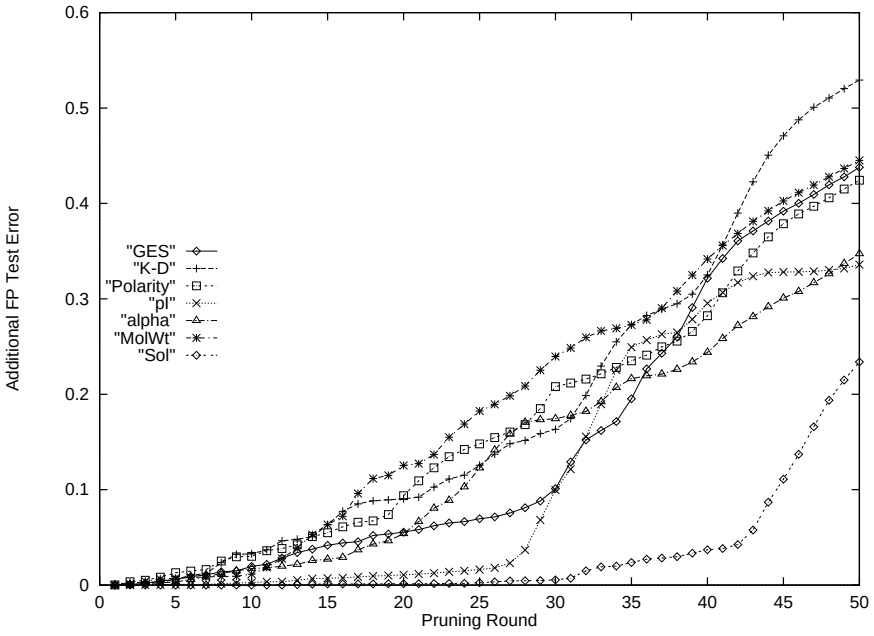

Figure 20. Additional false positive error on the test set versus pruning round for Half on the protein data sets. Plots for train set error are very similar.

affinity, then the binding energy of B_i to t is

$$E_{B_i} = \min_{B_{ij} \in B_i} \left\{ \sum_{k=1}^{n} s_k V\left(t_k - B_{ijk}\right) \right\} \ ,$$

where

$$V(r) = 4\epsilon \left(\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^{6} \right)$$

is called the Lennard-Jones potential with parameters ϵ and σ, and r is the distance between the bag instances and the target point (we assume $r > \sigma$, so $V(r) < 0$). Finally, Amar et al.'s real-valued label for B_i is E_{B_i}/E_{max}, where $E_{max} = -\epsilon \sum_{k=1}^{n} s_k$. Thresholding at 1/2 converts it to a binary label.

In the generalization we used, there are multiple target points ("subtargets"). Now the label of bag B_i is positive iff each subtarget induces a value $\geq 1/2$ in some point B_{ij} in the bag. That is, we threshold

$$\min_{t \in T} \left\{ \max_{B_{ij} \in B_i} \left\{ \frac{\sum_{k=1}^{n} s_k \, V\left(t_k - B_{ijk}\right)}{E_{max}} \right\} \right\} \ , \tag{4}$$

which is a half-Hausdorff *similarity* measure, analogous to the dissimilarity measure of Equation (2).

We used Amar et al.'s modified data generator to build two different types of data. The first was 4-dimensional data with 4 subtargets ("4/4"), and the second was 5-dimensional data with 4 subtargets ("5/4"). For each type, we ran 10 experiments: we first randomly generated 4 subtargets and then created three sets of bags, each of size 200. One bag was for training all algorithms, one bag was for testing, and the third was used by EMDD as a validation set for selecting its final hypothesis (so EMDD had a *de facto* training set of 400 bags). Each of the 4/4 data sets consisted of approximately 20% positive bags, and each of the 5/4 sets consisted of approximately 47% positive bags (the difference came from how the s_k values were set and the difference in dimensionality). Finally, since the data's dimensionality would make the number of groups prohibitively large, we k-means clustered the data and used the clusters' point representatives to build our algorithm's groups. I.e. the original data was still used, but Winnow could not distinguish between any points in the same region of Figure 2. The 4/4 data was clustered into $m = 8$ clusters and the 5/4 data had $m = 5$ clusters. EMDD and DD used the original (unclustered) data, so they had much more information on the data set.

Results are presented in Table 4. Interestingly, even though Equation (4) defines a target concept that can be represented by Half, we found that Half often did not reach 100% accuracy on the training set within 300 rounds. When it did, it required 4 times as many training rounds and still had FN

Table 4. Results on the affinity data sets.

Set	Full		EMDD		DD	
	FP	FN	FP	FN	FP	FN
4/4	0.102	**0.377**	*0.038*	0.843	0.125	0.520
5/4	**0.201**	0.224	0.263	*0.109*	0.274	0.108

error much higher than Full. As in Section 6.2.1, we believe that this is due to the reduced resolution caused by clustering. Thus to save time we focussed on Full, which converged in an average 25 rounds for both 4/4 and 5/4. Boldface numbers indicate where Full's advantage over both EMDD and DD is significant at at least 95% using a paired t test, and italicized numbers indicate where EMDD's advantage over Full is 95% significant. Further, Full's 4/4 FP error is less than DD's at 95%, and DD's FN error on 5/4 was lower than Full's at 75%.

We suspect that some of Full's error (and Half's poorer performance) comes from the data's low resolution due to clustering in forming the grid. We corroborated this hypothesis by rerunning EMDD and DD on data remapped as in Section 6.2, which maps each point $\vec{x}$ to its region's center $\vec{c}_{\vec{x}}$. For the 5/4 data, EMDD's FP error increased to 0.301 (an increase that is significant at 99%) and its FN error increased to 0.114 (significant at 60%). DD's FP error stayed the same and its FN error increased to 0.135 (significant at 95%). Further, for 4/4, DD's FP error increased to 0.177 (significant at 85%) and its FN error decreased slightly, though not significantly. EMDD's FN error increased to 0.945 (significant at 85%), but its FP error *decreased* to 0.022 (significant at 75%). Aside from the single decrease in FP error (which came at a costly increase to FN error), there is an obvious degradation in performance for EMDD and DD when using the clustered data on this problem. Thus had our algorithms had more information, they likely would generalize better, and it is also possible that Half would improve past Full, given Equation (4).

We tested the effect of group pruning by reducing the number of groups by 10% each round for 50 rounds. The results are summarized in Figures 21 and 22. Figure 21 shows the average cumulative fraction of the weight that is pruned in each round for Full. We see that before round 40, very little weight was pruned from any hypothesis, despite having pruned $1 - 0.9^{40} > 0.98$ of the groups, implying that relatively few of Winnow's inputs are relevant. After this point, the fraction of weight pruned increased faster. Not surprisingly, this corresponds to a faster increase in FP error, as shown in Figure 13. Interestingly though, the additional error even after 50 rounds was much less than for all our other experiments except sunset.

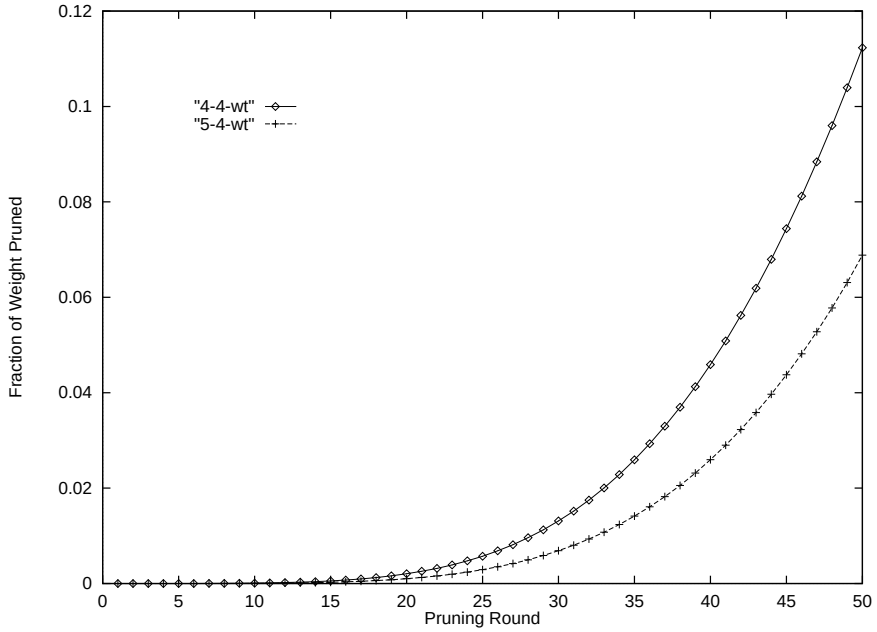

Figure 21. Fraction of weight pruned versus pruning round for Full on the affinity data.

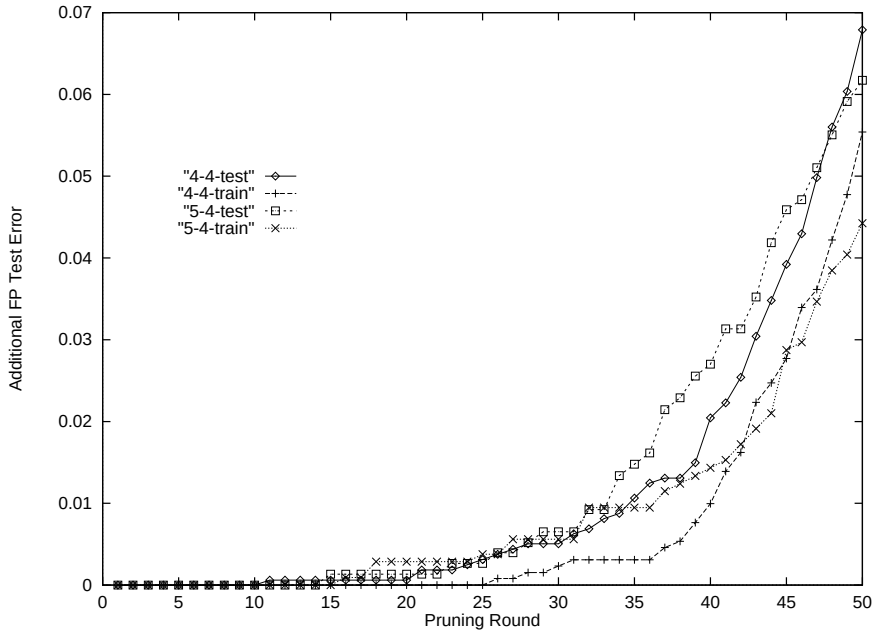

Figure 22. Additional false positive error on the testing and training sets versus pruning round for Full on the affinity data.

7. Conclusions and Future Work

While the standard MIL model is powerful, there exist applications with natural target concepts that cannot be represented. We cast the well-studied concept classes of geometric patterns into a more general MIL framework and proposed extensions to reflect requirements of real pattern recognition problems. We then described several application areas of this new MIL model. Finally, we modified a learning-theoretic algorithm for learning geometric patterns to fit our more general model and presented the first empirical results of this algorithm. Our results support our claim that many applications exist for which generalized MIL better captures the requirements than the conventional model.

Since Half's hypothesis space is a subspace of Full's, we expected Half to generalize better than Full on learning tasks that seem to naturally fit half-Hausdorff since it should be able to avoid overfitting more easily. What we discovered was quite different, however. Half often could not achieve 100% accuracy on the training sets like Full did, even when given 5–10 times the number of training rounds. Experiments in which Half did not achieve 100% training accuracy yielded hypotheses with very high FN error. Further, when Half did reach 100% accuracy on the training set, its generalization error was at best indistinguishable from (and at worst higher than) Full's. There is some evidence (Section 6.2.1) that Half is hurt more by clustering the training data than Full is. If this is true, then it would explain why Full outperforms Half on sunset (Section 6.2.1) and antagonist drugs (Section 6.4). The results on conjunctive CBIR are better explained by the nature of the target concept, and Full's advantage over Half on the robot data is likely due to the existence of other information in the data that Full can exploit and Half cannot.

In ongoing work, we are scaling our algorithm to handle more dimensions via two general methods. First, we are exploring new ways to build the set of groups besides partitioning the space into regions as done by Goldman et al.'s original algorithm (Figure 2). While straightforward, their technique is wasteful in that several distinct groups are defined that in fact contain the same subset of points. In particular, if only m points are in the grid, there are at most 2^m groups that contain distinct subsets of the m points, as opposed to the construction in Section 3.1 whose bound is $O(m^{2d+1})$. Thus by defining the groups differently, we can scale our algorithm to higher dimensions so long as m is small. It is unlikely that a group definition exists that is polynomial in both d and m, since this would yield an efficient algorithm for learning DNF formulas, which is unlikely to exist (Goldman et al., 2001).

A second optimization we are working on is using a fast approximation scheme to estimate the weighted sums of Winnow[22], including methods used

[22] Fortunately, extremely accurate estimates of the weighted sums are not necessary, so long as the estimates are on the same side of the threshold as the true weighted sum.

by Chawla et al. (2003) using Markov chain Monte Carlo methods. We will combine this with the new group definition to yield a new algorithm that we can use to better measure the effect of clustering the training data. We are also exploring techniques to merge unpruned groups that are near each other, to further reduce their number. This should produce hypotheses that are easily interpretable by humans, hopefully consisting of only a few 10s or 100s of groups.

Another way to expand this work is to find a kernel to quickly compute the weighted sum of the inputs, entirely eliminating the need to enumerate the groups, yielding a more scalable algorithm. The down side to this is that Winnow can no longer be used as the core algorithm. Since most of the features are irrelevant (as shown by our group pruning experiments), the speedup of multiplicative weight update algorithms like Winnow over additive weight update algorithms like the Perceptron is significant (Kivinen and Warmuth, 1997).

Finally, there is the open problem of developing other algorithms in this model. Certainly DD and EMDD are candidates to extend into this model, though care must be taken to keep them efficient, especially if one wants to represent a ranked version of full- or half-Hausdorff (requiring a hypothesis to capture the notion of an r-of-k threshold function) or to define a hypothesis that requires a positive bag to have no points in a particular region, like our "field and not sky" task. Also, Goldman and Scott (in press) developed an algorithm analogous to Goldman et al. (2001) that handles real-valued labels rather than binary ones. It would be interesting to apply this algorithm to problems where real-valued labels are useful (Dooly et al., 2002; Ray and Page, 2001), though computational complexity would be an issue.

Acknowledgments

The authors thank Corey Maley, Yixin Guo, Chad Michel, and Cory Strope for help with the implementation. They also thank Brian Pinette for the robot data, Etsuko Moriyama, Peggy Wen, Haifeng Ji, Vadim Gladyshev, Gregory Kryukov, and Dimitri Fomenko for the protein data, 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 affinity data generator, Qingping Tao for his comments on a draft of this paper, and Bruce Draper and Paul Rybski for their discussions. This research was funded in part by NSF grants CCR-9877080 (with matching funds from CCIS), 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 research was completed in part utilizing UNL's Research Computing Facility. Josh

Brown performed this work in the Dept. of Computer Science, University of Nebraska.

References

Andrews, S., I. Tsochantaridis, and T. Hofmann: 2002, 'Support vector machines for multiple-instance learning'. In: *Advances in Neural Information Processing Systems 15*.

Auer, P.: 1997, 'On learning from multi-instance examples: Empirical evaluation of a theoretical approach'. In: *Proc. 14th International Conference on Machine Learning*. pp. 21–29, Morgan Kaufmann.

Auer, P., P. M. Long, and A. Srinivasan: 1997, 'Approximating hyper-rectangles: Learning and pseudo-random sets'. In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*. pp. 314–323, ACM.

Ballard, D. H. and C. M. Brown: 1982, *Computer Vision*. Prentice Hall.

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

Chawla, D., L. Li, and S. D. Scott: 2003, 'On Approximating Weighted Sums with Exponentially Many Terms'. Technical Report TR03-02-02, University of Nebraska. Early version in COLT '01.

Daubechies, I.: 1988, 'Orthonormal Bases of compactly supported wavelets'. *Comm. Pure. and Appl. Math.* **41**, 909–996.

De Raedt, L.: 1998, 'Attribute-Value Learning versus inductive logic programming: The missing links'. In: *Proc. 8th International Conference on Inductive Logic Programming*. Springer Verlag.

Dietterich, T. G., R. H. Lathrop, and T. Lozano-Perez: 1997, 'Solving the Multiple-Instance Problem with Axis-Parallel Rectangles'. *Artificial Intelligence* **89**(1–2), 31–71.

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

Durbin, R., S. Eddy, A. Krogh, and G. Mitchison: 1998, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

Follmann, H. and I. Haberlein: 1995, 'Thioredoxins: universal, yet specific thiol-disulfide redox cofactors'. *Biofactors* **5**, 147–156.

Goldberg, P. W. and S. A. Goldman: 1994, 'Learning one-dimensional geometric patterns under one-sided random misclassification noise'. In: *Proc. 7th Annu. ACM Workshop on Comput. Learning Theory*. pp. 246–255, ACM Press, New York, NY.

Goldberg, P. W., S. A. Goldman, and S. D. Scott: 1996, 'PAC Learning of One-Dimensional Patterns'. *Machine Learning* **25**, 51–70.

Goldman, S. A., S. K. Kwek, and S. D. Scott: 2001, 'Agnostic learning of geometric patterns'. *Journal of Computer and System Sciences* **6**(1), 123–151.

Goldman, S. A. and S. D. Scott: 1999, 'A Theoretical and Empirical Study of a Noise-Tolerant Algorithm to Learn Geometric Patterns'. *Machine Learning* **37**(1), 5–49.

Goldman, S. A. and S. D. Scott: in press, 'Multiple-instance learning of real-valued geometric patterns'. *Annals of Mathematics and Artificial Intelligence*. Early version in technical report UNL-CSE-99-006, University of Nebraska.

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

Hong, J., X. Tan, B. Pinette, R. Weiss, and E. Riseman: 1992, 'Image-based Homing'. *IEEE Control Systems Magazine* **12**(1), 38–45.

Huttenlocher, D. P., G. A. Klanderman, and W. J. Rucklidge: 1993, 'Comparing images using the Hausdorff distance'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, 850–863.

Huttenlocher, D. P. and W. J. Rucklidge: 1992, 'A multi-resolution technique for comparing images using the Hausdorff distance'. Technical Report 92-1321, Dept. of Computer Science, Cornell University.

Kim, J., E. N. Moriyama, C. G. Warr, P. J. Clyne, and J. R. Carlson: 2000, 'Identification of novel multi-transmembrane proteins from genomic databases using quasi-periodic structural properties'. *Bioinformatics* **16**(9), 767–775.

Kivinen, J. and M. K. Warmuth: 1997, 'Exponentiated gradient versus gradient descent for linear predictors'. *Information and Computation* **132**(1), 1–63.

Levitt, T. and D. Lawton: 1990, 'Qualitative navigation for mobile robots'. *Artificial Intelligence* **44**(3), 305–360.

Littlestone, N.: 1988, 'Learning Quickly When Irrelevant Attributes Abound: A New Linear-threshold Algorithm'. *Machine Learning* **2**, 285–318.

Long, P. M. and L. Tan: 1998, 'PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples'. *Machine Learning* **30**, 7–21.

Maron, O.: 1998, 'Learning from Ambiguity'. Ph.D. thesis, Dept. of Electrical Engineering and Computer Science, M.I.T.

Maron, O. and T. Lozano-Pérez: 1998, 'A framework for multiple-instance learning'. In: *Advances in Neural Information Processing Systems 10*.

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

Martin, J.: 1995, 'Thioredoxin—a fold for all reasons'. *Structure* **3**, 245–250.

Pinette, B.: 1993, 'Image-Based Navigation Through Large-Scaled Environments'. Ph.D. thesis, University of Massachusetts, Amherst.

Ramon, J. and L. D. Raedt: 2000, 'Multi Instance Neural Networks'. In: *Proceedings of the ICML-2000 Workshop on Attribute-Value and Relational Learning*.

Ray, S. and D. Page: 2001, 'Multiple instance regression'. In: *Proceedings of the Eighteenth International Conference on Machine Learning*. pp. 425–432.

Ruffo, G.: 2000, 'Learning single and multiple instance decision trees for computer security applications'. Ph.D. thesis, Dept. of Computer Science, University of Turin, Torino, Italy.

Scott, S.: 2000, 'Geometric Patterns: Algorithms and Applications'. In: *Proceedings of the ICML 2000 Workshop on Machine Learning of Spatial Knowledge*. pp. 109–115.

Scott, S. D., H. Ji, P. Wen, D. E. Fomenko, and V. N. Gladyshev: 2003, 'On modeling protein superfamilies with low primary sequence conservation'. Technical Report TR-UNL-CSE-2003-4, Department of Computer Science, University of Nebraska.

Sim, D.-G., O.-K. Kwon, and R.-H. Park: 1999, 'Object matching algorithms using robust Hausdorff distance measures'. *IEEE Transactions on Image Processing* **8**(3), 425–429.

Stryer, L.: 1995, *Biochemistry*. W. H. Freeman and Company, fourth edition.

Suzuki, H. and S. Arimoto: 1988, 'Visual control of autonomous mobile robot based on self-organizing model for pattern learning'. *Journal of Robotic Systems* **5**(5), 453–470.

Wang, J. and J. D. Zucker: 2000, 'Solving the Multiple-Instance Problem: A Lazy Learning Approach'. In: *Proc. 17th International Conf. on Machine Learning*. pp. 1119–1125.

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

Zhang, Q. and S. A. Goldman: 2001, 'EM-DD: An improved multiple-instance learning technique'. In: *Neural Information Processing Systems 14*.

Zhang, Q., S. A. Goldman, W. Yu, and J. E. Fritts: 2002, 'Content-Based image retrieval using multiple-instance learning'. In: *Proc. 19th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA.