Supervised versus Multiple Instance Learning: An Empirical Comparison

Soumya Ray SRAY@CS.WISC.EDU
Mark Craven CRAVEN@BIOSTAT.WISC.EDU
Departments of Computer Science and Biostatistics & Medical Informatics, University of Wisconsin, USA

Abstract

We empirically study the relationship between supervised and multiple instance (MI) learning. Algorithms to learn various concepts have been adapted to the MI representation. However, it is also known that concepts that are PAC-learnable with one-sided noise can be learned from MI data. A relevant question then is: how well do supervised learners do on MI data? We attempt to answer this question by looking at a cross section of MI data sets from various domains coupled with a number of learning algorithms including Diverse Density, Logistic Regression, nonlinear Support Vector Machines and FOIL. We consider a supervised and MI version of each learner. Several interesting conclusions emerge from our work: (1) no MI algorithm is superior across all tested domains, (2) some MI algorithms are consistently superior to their supervised counterparts, (3) using high false-positive costs can improve a supervised learner's performance in MI domains, and (4) in several domains, a supervised algorithm is superior to any MI algorithm we tested.

1. Introduction

The Multiple Instance (MI) setting was introduced by Dietterich et al. (1997) in the context of drug activity prediction. In a multiple instance problem, instances are naturally organized into *bags* (i.e., *multisets*) and it is the bags, instead of individual instances, that are labeled for training. MI learners assume that every instance in a bag labeled *negative* is actually negative, whereas at least one instance in a bag labeled *positive* is actually positive. Note that a positive bag may contain negative instances.

Since its introduction, a wide variety of tasks have been formulated as multiple-instance learning problems. Among these tasks are content-based image retrieval (Maron & Ratan, 1998; Andrews et al., 2003), stock prediction (Maron, 1998), text classification (Andrews et al., 2003), and protein family modeling (Tao et al., 2004). Various algorithms have been introduced for learning in the MI setting. In their original work, Dietterich et al. (1997) described an algorithm to learn axis-parallel rectangles (APRs) from MI data. A framework called Diverse Density (Maron, 1998) has been proposed and used to learn Gaussian concepts. Others have proposed algorithms adapting Support Vector Machines (Andrews et al., 2003; Gartner et al., 2002) to this problem, either by changing the objective function or the kernel used. Some approaches using lazy learning (Wang & Zucker, 2000) and decision trees (Ruffo, 2000) have been investigated in this context as well. Recently, work has been done investigating the use of ensembles (Auer & Ortner, 2004) to learn multiple-instance concepts.

Although MI learning has been investigated in this wide range of problem domains with a wide range of approaches, most studies have empirically compared only a few approaches using only a few (often one) data sets. Because of the limited scope of these studies, there is not a clear sense of which multiple instance algorithms might be best suited to which domains. Does there exist a single MI algorithm well suited to every MI domain, or, even if not always the best, is consistently among the best algorithms for these tasks? We attempt to answer this question in our current work.

Another interesting question concerns the relationship between ordinary supervised and MI learning. In their work on the theory of learning from MI examples, Blum and Kalai (1998) show that a concept that is

Appearing in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005. Copyright 2005 by the author(s)/owner(s).

PAC-learnable with one-sided classification noise is PAC-learnable from MI examples. Put in an empirical context, this result can imply that, given enough examples, transforming an MI dataset into a standard supervised representation can allow a supervised learning algorithm to successfully learn the underlying instance concept. While the result is only applicable to PAC-learnable concepts, one may hypothesize that it holds for other concepts in practice. Thus, a relevant question is, how well do supervised algorithms learn from MI data in general? This question has not been addressed in previous research on MI learning.

Here, we address these questions by conducting an empirical evaluation in which we apply a wide variety of multiple-instance learning methods to a wide range of tasks that have been previously considered in the MI literature. Moreover, to address the question of how well supervised learning methods learn in MI domains, our experiments also evaluate a supervised-learning counterpart for every MI algorithm we consider. Each of these counterparts is able to represent and search exactly the same concept class as its MI partner.

2. Algorithms

In this section, we describe the algorithms we use in our experiments. **Diverse Density** is perhaps the best known framework for MI learning (Maron, 1998). The idea behind this approach is that, given a set of positive and negative bags, we wish to learn a concept that is "close" to at least one instance from each positive bag, while remaining "far" from every instance in every negative bag. Thus, the concept must describe a region of instance space that is "dense" in instances from positive bags, and is also "diverse" in that it describes every positive bag. More formally, let $DD(t) = \frac{1}{Z}\left(\prod_i \Pr(t|B_i^+)\prod_i \Pr(t|B_i^-)\right)$, where t is a candidate concept, B_i^+ represents the i^{th} positive bag, and B_i^- represents the i^{th} negative bag. We seek a concept that maximizes $DD(t)$. The concept generates the instances of a bag, rather than the bag itself. To score a concept with respect to a bag, we combine t's probabilities for instances using a function based on noisy-or (Pearl, 1988):

$$\Pr(t|B_i^+) \propto (1 - \prod_j (1 - \Pr(B_{ij}^+ \in t))) \quad (1)$$

$$\Pr(t|B_i^-) \propto \prod_j (1 - \Pr(B_{ij}^- \in t)) \quad (2)$$

Here the instances B_{ij}^+ and B_{ij}^- belonging to t are the "causes" of the "event" that "t is the target". The concept class investigated by Maron (1998) is the class of generative Gaussian models, which are parameterized by the mean μ and a "scale" $s = \frac{1}{2\sigma^2}$:

$$\Pr(B_{ij} \in t) \propto e^{-\sum_l (s_l(B_{ijl}-\mu_l)^2)},$$

where l ranges over features.

To apply a standard supervised learning algorithm to MI data, we could pick a point from a positive bag at random and call it positive, as suggested previously (Blum & Kalai, 1998). In practice, positive bags may have multiple positive instances. Thus, we convert MI data to a supervised sample by assuming that the label of a bag applies to each instance it contains.

A simple **Gaussian model** is the supervised learning counterpart of Diverse Density. This model maximizes $DD(t)$ with the noisy-or replaced by simple product. In this case, equation (1) is replaced by $\Pr(t|B_i^+) \propto \left(\prod_j \Pr(B_{ij}^+ \in t)\right)$, everything else remaining the same. Thus, this algorithm assumes that every instance in a positive bag is labeled positive. Observe that, since every instance in a negative bag is considered to be truly negative in the MI setting, we do not need to change the part of the objective function dealing with negative bags when translating the data to a supervised learning setting.

Diverse Density with k disjuncts is a variant of Diverse Density investigated by Maron in his work (1998). This is a class of disjunctive Gaussian concepts, where the probability of an instance belonging to a concept is given by the maximum probability of belonging to any of the disjuncts:

$$\Pr(B_{ij} \in \{t_1, \dots t_k\})) \propto \\ \text{softmax}_\alpha(\Pr(B_{ij} \in t_1), \dots, \Pr(B_{ij} \in t_k)),$$

where

$$\text{softmax}_\alpha(x_1, \dots, x_n) = \frac{\sum_{1 \le i \le n} x_i e^{\alpha x_i}}{\sum_{1 \le i \le n} e^{\alpha x_i}}$$

is a smooth approximation to the "max" function (the approximation improves as α is increased). In this work, we investigate Diverse Density with $k = 3$ and $k = 5$ disjuncts, abbreviated as **DD(3)** and **DD(5)**. For consistency, we abbreviate the original Diverse Density algorithm as **DD(1)**.

A **Statistic Kernel** has been proposed (Gartner et al., 2002) to adapt Support Vector Machines to the MI framework. This kernel is a function over bags rather than instances. It is defined by transforming a bag into a single feature vector and then applying a polynomial kernel to this representation. The specific transformation applied is as follows. Let $B_i = \{B_{i1}, B_{i2} \dots B_{in}\}$ be a bag of instances, and let each instance be described by a feature vector of length m. Then we define a feature vector of length $2m$ for the bag, where the j^{th} feature is the minimum of the j^{th} feature across every instance of the bag, and the $2j^{th}$ feature is the maximum of the j^{th} feature across every instance of the bag. Thus, this transform assumes that the discriminant will be a function of the extreme values of each feature in a bag, and not of the intermediate ones. Note that this is not a "true" multiple instance kernel, since the transformed feature vector is not a function of any single instance – in fact, each min/max feature value could be from a different instance in the bag. In our work, we use a quadratic kernel with this transformed feature vector.

A **Normalized Set Kernel** (NSK) has also been proposed (Gartner et al., 2002) for SVM classifiers applied to MI problems. This is defined as follows. Let κ be a kernel defined on the space of instances, and let X and Y be sets of instances. Then $k_{set}(X,Y) = \sum_{x \in X, y \in Y} \kappa(x,y)$ is a kernel on the sets X and Y. The normalized set kernel is defined as:

$$n(X,Y) = \frac{k_{set}(X,Y)}{\sqrt{k_{set}(X,X)} \cdot \sqrt{k_{set}(Y,Y)}}$$

In the MI representation, we can apply this kernel to bags of instances. The normalization factor is necessary to correct for varying bag sizes. While Gartner et al. use a Gaussian kernel for κ, we report results using a Quadratic kernel, to be consistent with the other SVM learners we use. Further, we have run our experiments using both and have found that the Quadratic kernel generally results in more accurate models.

For our supervised counterpart to the above, we use a simple **Quadratic Kernel** over instances in bags. As before, in this setting, we assume that every instance in a bag has the same label as that of the bag.

Relational learners like **FOIL** (Quinlan, 1990) can naturally handle the multiple instance representation. In our experiments, we apply FOIL to instances described in a single table of attribute-value pairs. We construct an MI representation for a task by defining a target relation over bags (for example, pos-bag(B)). Each positive bag is specified to satisfy the target, while no negative bag does so. An instance relation, instance(B,N), is then defined that describes each instance N in each bag B. This representation biases FOIL to learn rules of the form pos-bag(B) :- instance(B,N), properties-of-N. These rules represent an MI concept, where a bag is positive if any instance in it satisfies the learned properties. We can also use FOIL in a supervised setting by defining a target relation over instances. In particular, we define a target relation, pos-instance(N), and specify that every instance from a positive bag in our training set satisfies this relation, while no instance from a negative bag does. In either setting, in our domains, FOIL learns clauses similar to axis-parallel rectangles, where each literal defines an upper or lower bound on the value of some variable in the head of the clause (or in the instance literal). The same algorithm is used in both cases, but to distinguish the former representation from the latter in our experiments, we call the former **MI/FOIL**.

We have designed a novel algorithm, **Multiple Instance Logistic Regression**, to learn linear models in an MI setting. This algorithm is derived by generalizing the Diverse Density framework. In multiple instance classification, we seek $\Pr(y_i = 1 | B_i = \{B_{i1}, B_{i2} \ldots B_{in}\})$, the conditional probability of the label of the i^{th} bag being positive given the instances in the bag. If we are given a model that can estimate the equivalent probability for an instance, $\Pr(y_{ij} = 1 | B_{ij})$, we can use a *combining function*, such as softmax, to combine the posterior probabilities over the instances of a bag and obtain an estimate for the posterior probability $\Pr(y_i = 1 | B_i)$. Observe that, in this case, it is the combining function that encodes the multiple instance assumption. Thus, if one of the instances is very likely to be positive, as determined by the instance model, the combining function must be such that its estimate of the bag's positive-ness will be high. Further, observe that this general setting allows any model that can learn class conditional probabilities in a supervised setting to be used in a multiple instance setting as well. In our work, we use the **Logistic Regression** (**LR**) algorithm with parameters $(\mathbf{w}, b)$ to estimate conditional probabilities for each instance, and use softmax to combine these to obtain the conditional probabilities for each bag:

$$S_{ij} = \Pr(y_{ij} = 1 | B_{ij}) = \frac{1}{1 + e^{-(\mathbf{w} \cdot B_{ij} + b)}}$$
$$\Pr(y_i = 1 | B_i) = \text{softmax}_\alpha(S_{i1}, \ldots, S_{in})$$

We call this algorithm Multiple Instance Logistic Regression, abbreviated **MI/LR**. The supervised learning counterpart to this algorithm is ordinary Logistic Regression. A similar approach, that averages the instances probabilities to obtain bag probabilities, has been investigated previously (Xu & Frank, 2004).

3. Problem Domains

In this section, we give brief overviews of the domains we consider in our experiments, and how the multiple instance representation has been used in each case.

Drug activity was the motivating application for the multiple instance representation (Dietterich et al., 1997). In this domain, we wish to predict how strongly a given molecule will bind to a target. Each molecule is a three-dimensional entity and takes on multiple shapes or *conformations* in solution. We know that for every molecule showing activity, at least one its low energy conformations possesses the right shape for interacting with the target. Similarly, if the molecule does not show drug-like activity, none of its conformations possess the right shape for interaction. Thus, each molecule is represented as a bag, where each instance is a low energy conformation of the molecule. A well-known example from this domain that we use in our experiments is the MUSK dataset. The positive class in this data consists of molecules that smell "musky". This dataset has two variants, MUSK1 and MUSK2, both with similar numbers of bags, but with MUSK2 having many more instances per bag.

Content Based Image Retrieval (CBIR) is another domain where the MI representation has been used (Maron & Lozano-Pérez, 1998; Zhang et al., 2002). In this domain, the task is to find images that contain objects of interest, such as tigers, in a database of images. An image is represented by a bag. An instance in a bag corresponds to a segment in the image. The underlying assumption is that the object of interest is contained in (at least) one segment of the image. In our experiments, we use two sets of CBIR data. The first task is to separate three categories of animals, Tiger, Elephant, and Fox, from background images (Andrews et al., 2003). The second task is to separate natural scenes with specific features from others (Zhang et al., 2002). We use the categories of Flower, Sunset and Waterfall from this set. Negative examples for each category consist of images from the other categories as negative examples, along with images from the categories Mountain and Field.

The **identification of TrX proteins** has recently been framed as a multiple instance problem (Tao et al., 2004). The objective is to classify given protein sequences according to whether they belong to the family of TrX proteins. The given proteins are first aligned with respect to a motif that is known to be conserved in members of the family. Each aligned protein is represented by a bag. A bag is labeled positive if the protein belongs to the family, and negative otherwise. An instance in a bag corresponds to a position in a fixed length sequence around the conserved motif. Each position is described by properties of the amino acid at that position, and smoothed using the same properties from its 16 neighbors.

Text Categorization is the final domain that we consider that has used the MI representation. In our experiments, we use a novel data set for this task obtained as part of Task 2 of the BioCreative Text Mining Challenge (Blaschke et al., 2005). In this task, we are given full-text articles from biomedical journals and the names of human proteins. The objective is to label each article and protein with codes from the Gene Ontology (GO). The GO consists of three hierarchical domains of standardized biological terms referring to cellular components, biological processes and molecular functions. Each such term is mapped to a unique "GO code". A <protein, article> pair is labeled with a GO code if the article contains text that links the protein to the component, process or function described by the GO code. An example of the task is shown in Figure 1. We develop a two-stage system to solve this task. The first stage identifies all passages in the text that contain the protein of interest and some weak evidence about an arbitrary GO code. In the second stage, we learn a model to predict how likely it is that a document actually relates the protein to the GO code, given the output of the first part. To learn such a model, we could assume that every passage in a training document that mentions the protein and some text supporting the GO code also relates the protein to the GO code. However, this is not a realistic assumption. Usually, in a training document annotated with a GO code C for a protein P, there exist *some* passages that link C to P, but not every passage that mentions P and C does so. This can be formulated as a multiple instance problem as follows. Positive bags for our model consist of documents that are labeled with GO codes. Each instance in a bag is a paragraph in the document, output by the first stage of our system. Each paragraph is described by a set of word count features, along with a set of numerical features that capture some aspects of the protein-GO code interaction, such as the average distance between mentions of the protein and the evidence text for the GO code. Using this representation, we build three data sets for our experiments: one each for the Component, Function and Process domains in GO.

Protein: Mitochondrial 28S ribosomal protein S14
Article: PUBMED ID 10938081
...Three of the four currently identified mammalian mitochondrial small subunit ribosomal proteins that have prokaryotic homologs (S7, S10, and S14) are located in the head of the small subunit...

$\Downarrow$

Gene Ontology Code: GO:0003735
(structural constituent of ribosome)

Figure 1. An example of the BioCreative task. Given a protein name and the text of an article, annotate the protein with Gene Ontology codes the article has evidence for.

4. Experimental Methodology

To optimize the objective functions for the Diverse Density and Logistic Regression models, we use the BFGS method (Fletcher, 1980). Since the solutions to these objectives are usually only locally optimal, we restart each algorithm 10 times for each run. The initial parameter settings for the Diverse Density algorithm are chosen to model instances in positive bags (Maron, 1998). For the Logistic Regression algorithms, the initial parameter settings are chosen uniformly at random in $(0, 1)$. When the softmax function is used, the parameter α is set to 3.

The MI SVM kernels are implemented in the framework of Smooth Support Vector Machines (Lee & Mangasarian, 2001). For the supervised SVM, we used SVM^{light} (Joachims, 1999). SVM^{light} is run with the quadratic kernel, and default parameters otherwise.

FOIL is modified to use more than 52 variables (the default maximum limit), and is allowed to use up to 255 variables (this is much larger than the dimension of any of our data sets). On the TrX data set, FOIL is called with the "accuracy" parameter ("–a", which actually measures precision) set to 50. Since this data set has very few positive examples, FOIL frequently returns the empty clause without this modification.

We test these algorithms using 10-fold stratified cross validation on all data sets except the three BioCreative data sets. Every algorithm is given the same set of folds for each data set. Folds are constructed on bags, so that every instance in a given bag appears in the same fold. For the BioCreative data sets, the data is naturally partitioned into two sets, corresponding to articles published in the *Journal of Biomedical Chemistry* (*JBC*) and those published in *Nature*. In our experiments, we use the *JBC* articles to learn our models and the *Nature* articles to test them.

To evaluate the behavior of the algorithms, we construct Receiver Operating Characteristic (ROC) curves. These curves measure the true-positive rate of a classifier versus its false-positive rate as a threshold is varied across a measure of confidence in its predictions. To construct ROC graphs from our experiments, we pool the predictions of the algorithms across all folds. The supervised learning algorithms in our experiments generate independent predictions for each instance in a bag. To generate bag-level predictions from the output of these algorithms, we take the instance prediction made at the highest confidence to be the prediction for the bag. For confidence measures, we use conditional probability estimates of $\Pr(y_i = 1|B_i)$ for Diverse Density and Logistic Regression. For the SVM classifiers, we use the (signed) margin as a measure of confidence in the model's predictions. For the FOIL algorithm, we modify it to associate a confidence with each learned clause. This confidence is an m-estimate of the precision of the learned clause (Lavrac & Dzeroski, 1994). A test instance or bag is associated with the confidence of the first rule that covers it, or 0 if no rule covers it and it is predicted to be negative.

Due to lack of space, we cannot show all the ROC graphs for all experiments in this paper. In Figure 2 we show the ROC graphs obtained for the MUSK1 and Tiger data sets. In Table 1 we report a summary statistic, the area under the ROC graph (AROC) for every data set and algorithm (Bradley, 1997).

5. Discussion of Experiments

From the results in Table 1, we can draw several interesting conclusions.

Different inductive biases are appropriate for different MI problems. No single MI algorithm dominates over the others across all data sets. Not surprisingly, algorithms with different hypothesis-space biases are suitable for different domains. Thus, Gaussian concepts perform the best in two domains, linear concepts in four, quadratic concepts in five and rule-based concepts in one domain. This result suggests that when addressing an apparent MI problem, one should investigate a variety of learning approaches.

Ordinary supervised learning algorithms learn accurate models in many MI settings. While no single algorithm dominates overall, we observe that in general, supervised learning algorithms do well on many MI data sets. In fact, for several of the MI data sets we consider, a supervised learning algorithm is the best overall. For example, the Quadratic kernel SVM has the best AROC on three data sets, FOIL working with a supervised representation on one data set, and Logistic Regression on one data set. Further, the SVM with the Statistic kernel is the best on three data sets. As we noted earlier, this is not a "true MI kernel" in some sense. Because it allows every instance in a bag to contribute equally to the transformed feature vector, it implicitly assumes that the bag's label applies to every instance within it. Thus, we consider this kernel to be a variant of a supervised kernel. Further, observe that on every data set where the Statistic kernel does well, the Quadratic kernel (ordinary supervised learner) also tends to do well, though the reverse is not true. Finally, we observe that, on the data sets where MI algorithms are the best, a supervised learning algorithm is often very close. This is the case for

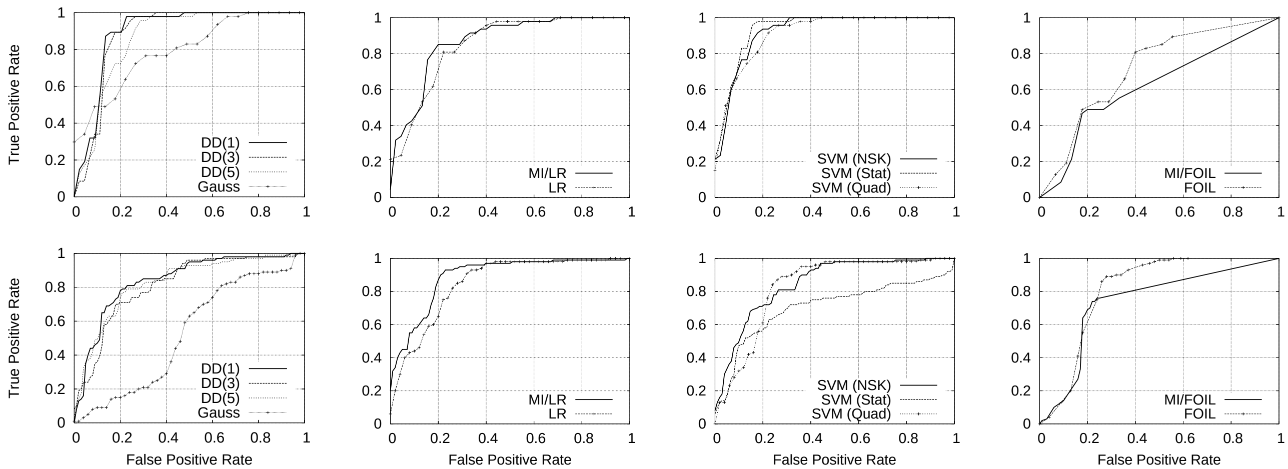

Figure 2. ROC graphs for the MUSK1 data set (top) and the Tiger data set (bottom). Each graph shows the ROC curve for a set of related models. They are, left to right: Diverse Density with 1, 3 and 5 disjuncts and the Gaussian model; Logistic Regression and Multiple Instance Logistic Regression; SVMs with the Normalized Set kernel, Statistic kernel, and Quadratic kernel; FOIL and MI/FOIL.

Table 1. Area under ROC curves for each method on each data set. Supervised methods are in italics. Bold values indicate the best AROC for each data set.

Data	DD(1)	DD(3)	DD(5)	*Gauss*	MI/LR	*LR*	SVM (Stat)	SVM (NSK)	*SVM (Quad)*	MI/FOIL	*FOIL*
Musk1	0.895	0.883	0.861	0.795	0.867	0.847	**0.937**	0.924	0.903	0.621	0.719
Musk2	**0.903**	0.850	0.838	0.850	0.870	0.837	0.892	0.866	0.847	0.725	0.765
Tiger	0.841	0.814	0.828	0.525	**0.890**	0.849	0.705	0.848	0.827	0.737	0.806
Elephant	0.907	0.892	0.902	0.734	0.933	0.931	0.856	0.915	**0.944**	0.821	0.859
Fox	0.631	0.639	0.656	0.554	0.633	0.593	0.618	0.525	0.579	0.655	**0.696**
Flower	0.878	0.879	0.876	0.603	0.919	0.899	0.874	0.901	**0.953**	0.831	0.933
Sunset	0.878	0.878	0.878	0.645	0.909	0.899	**0.979**	0.970	0.969	0.841	0.946
Waterfall	0.857	0.875	0.866	0.581	0.926	0.928	**0.950**	0.944	**0.950**	0.776	0.915
TrX	0.805	0.797	**0.828**	0.340	0.752	0.720	0.550	0.716	0.584	0.543	0.721
Component	0.724	0.720	0.703	0.683	**0.867**	0.849	0.745	0.724	0.752	0.686	0.744
Function	0.743	0.749	0.748	0.694	0.837	**0.863**	0.631	0.738	0.800	0.736	0.766
Process	0.820	0.809	0.816	0.792	**0.847**	0.823	0.742	0.787	0.807	0.766	0.792

MI/LR and LR, for example, on the BioCreative data. Similarly, while Diverse Density has the best AROC on MUSK2, the Statistic Kernel does almost as well.

What could explain the relative success of ordinary supervised learning methods in these domains? While this question needs to be explored further, we can offer some intuitive ideas. First, it may be the case that positive bags frequently have multiple "true-positive" instances (i.e., instances that truly are positive). Recall, the MI assumption is that each positive bag contains *at least one* instance in each positive bag. Many MI algorithms are biased to model positive bags as containing *nearly one* positive instance each. For example, with the noisy-or and softmax combining functions, the incremental benefit of "covering" additional instances in a positive bag decreases exponentially with number covered. Thus, a supervised learner may have a more appropriate bias than an MI learner in domains in which positive bags contain a relatively high density of positive instances because the supervised learner's objective function changes uniformly as more instances are classified according to the labels of their bags.

A second possibility is that the nature of the "negative" instances in the positive bags (i.e. false-positives) may be different from the nature of the negative instances in negative bags. For example, in the drug activity domain, false-positives are inactive conformations of active molecules. Instances in negative bags are conformations of inactive molecules. It is possible that, even though the false-positive conformations are inactive, they are more similar to the active conformations than the conformations of negative molecules. Now if conformations like these are produced *only by active molecules*, then misclassifying them carries no penalty as long as no similar instances are produced by inactive molecules.

We hypothesize that it may be possible to use differential misclassification costs to improve the accuracy of supervised learners in MI domains. Since we expect to encounter a large number of false positives (i.e., negative instances from positive bags) in MI domains, we can modify the objective function of a supervised learning algorithm to introduce a higher cost for false positives (FPs) than for false negatives. A "false negative" in this context refers to an instance from a positive bag that is classified as negative; such an instance could well be a true negative with respect to the underlying target concept. We conduct an experiment in which we use six data sets where the MI version of Logistic Regression has accuracy superior to the supervised version when the misclassification costs are uniform (see Table 1). For each data set, we run standard LR with FP cost multipliers ranging from 1 to 10. Let $L(x)$ denote the area under ROC for LR when the cost multiplier is x. $L(1)$ is therefore the same as standard LR. Let M denote the area under ROC for MI/LR. Then for each dataset and multiplier x, we plot the quantity $\left(\frac{L(x)-L(1)}{M-L(1)}\right)$. This quantity measures the fraction of the difference in area between MI/LR and LR that is "recovered" by varying the cost multiplier. If this quantity is positive for some multiplier x, then LR with that multiplier does better than standard LR. If the quantity reaches 1 for some x, then LR with cost multiplier x achieves the same AROC as MI/LR. Figure 3 shows the results of this experiment. From this figure, we observe that for two data sets, Musk1 and TrX, increasing the cost multiplier results in models that equal the predictive accuracy of MI/LR. For Musk1, the final AROC for LR with a cost multiplier of 10 exceeds the AROC of MI/LR. For three other data sets, Musk2, Tiger and Fox, the area also improves as the false positive cost is increased (about 25% of the initial difference is recovered); however, the models constructed by MI/LR remain more accurate for all cost multipliers. Finally, we observe that on the Flower data set, using any cost multiplier greater than 1 results in a decrease in accuracy. Thus, in five out of six cases, we can improve the accuracy of the supervised models by tuning false positive costs, and in two cases we are able to equal or exceed the accuracy of the MI models by using this technique.

Some MI algorithms learn consistently more accurate models than their supervised-learning counterparts. Though supervised algorithms in general do well on MI data sets, if we restrict our attention to comparisons between each MI algorithm and its supervised

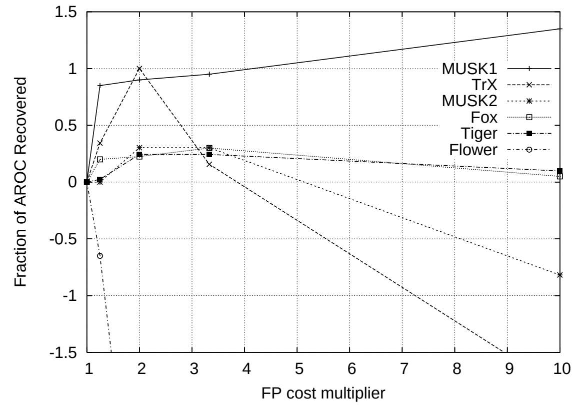

Figure 3. Effect of increasing the cost of false positives for the Logistic Regression algorithm. On the x-axis is the multiplier for the false positive cost (1 represents equal FP and FN costs). The y-axis shows the fraction of AROC difference between MI/LR and LR at FP cost=1 that is recovered as the cost changes. Thus, a value of 1 means that LR and MI/LR have the same AROC.

counterpart, we obtain a more mixed picture. Thus, Diverse Density is always superior to the supervised Gaussian model. Similarly, the MI/LR algorithm is often superior to the LR algorithm (9 out of 12 data sets). Therefore, for these algorithms, taking the MI setting into account when learning from MI data is usually useful. In the case of SVMs, on the other hand, there is not a clear winner between the supervised and the MI methods. With FOIL, the MI representation is always worse than its supervised learning counterpart. The MI representation for FOIL suffers from low recall (recall is identical to the true positive rate). This can be seen in the graphs in Figure 2. Since FOIL's "combining function" is an absolute disjunction, there is no gain in modeling more than one instance from any bag. This bias prevents it from trying to cover as many instances in a positive bag while still not covering any negatives, and leads to lower recall on the test set. This also indicates that there is often more than one positive instance in any positive bag.

Multiple-Instance Logistic Regression is a competitive method. Finally, we observe that the algorithm we have introduced in this work, MI Logistic Regression, performs quite well on many data sets. In data sets where it is not the best performer, it is almost always among the three best algorithms.

6. Conclusion

We have empirically evaluated the applicability of ordinary supervised learning algorithms to MI problems. We observe that different MI approaches are suited to different domains and some MI algorithms are always superior to their supervised counterparts. However, ordinary supervised learning algorithms also do well on many MI domains, and sometimes are the best algorithms for a task. Further, using higher false positive costs with supervised algorithms can improve their performance in MI domains. We have also introduced a novel MI algorithm that adapts Logistic Regression to MI data, and shown that it is competitive with other MI algorithms on our tasks.

Where then are algorithms with explicitly encoded MI biases relevant? Such algorithms may be useful in at least two situations which we have not investigated. First, it may be necessary not merely to be able to classify a bag correctly, but to *return a positive instance from a positive bag*. For example, in the drug activity domain, the algorithm may need to provide insight to the chemist about the structure of active conformations. Second, such algorithms may also be useful in domains with very sparse data. For example, Zhang et al. (2002) have investigated the task of retrieving images from a database assuming the target represents images a user has labeled as "interesting" or not. In this case, the algorithm may have to deal with a training set of only a few bags. If the bags contain many false positives, it is likely that an explicit MI bias will help the learning algorithm. These directions may prove fruitful in future work.

Acknowledgments

This research was supported in part by NIH grant R01-LM07050-01 and NSF grant IIS-0093016. We thank Stuart Andrews, Sally Goldman, Thomas Hofmann, Rouhollah Rahmani, Stephen Scott and Qingping Tao for kindly donating their data sets.

References

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2003). Support vector machines for multiple-instance learning. In *Advances in neural information processing systems*, vol. 15. MIT Press.

Auer, P., & Ortner, R. (2004). A boosting approach to multiple instance learning. *Proc. of the 15th European Conf. on Machine Learning* (pp. 63–74). Springer-Verlag.

Blaschke, C., Andres Leon, E., Krallinger, M., & Valencia, A. (2005). Evaluation of BioCreAtIvE assessment of task 2. *BMC Bioinformatics*, *6(Suppl. 1)*.

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

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*, 1145–1159.

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

Fletcher, R. (1980). *Practical methods of optimization*, vol. 1: Unconstrained Optimization, chapter 3. Wiley.

Gartner, T., Flach, P. A., Kowalczyk, A., & Smola, A. J. (2002). Multi-instance kernels. *Proc. of the 19th International Conf. on Machine Learning* (pp. 179–186). Morgan Kaufmann.

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*. MIT Press.

Lavrac, N., & Dzeroski, S. (1994). *Inductive logic programming: Techniques and applications*. Ellis Horwood.

Lee, Y.-J., & Mangasarian, O. L. (2001). SSVM: A smooth support vector machine. *Computational Optimization and Applications*, *20*, 5–22.

Maron, O. (1998). *Learning from ambiguity*. Doctoral dissertation, Dept. of EECS, MIT, Cambridge, MA.

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

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

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning*, *5*, 239–2666.

Ruffo, G. (2000). *Learning single and multiple instance decision trees for computer security applications*. Doctoral dissertation, Università di Torino, Torino, Italy.

Tao, Q., Scott, S. D., & Vinodchandran, N. V. (2004). SVM-based generalized multiple-instance learning via approximate box counting. *Proc. of the 21st International Conf. on Machine Learning* (pp. 779–806). Morgan Kaufmann.

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

Xu, X., & Frank, E. (2004). Logistic regression and boosting for labeled bags of instances. *Proc. of the Pacific-Asia Conf. on Knowledge Discovery and Data Mining*. Springer-Verlag.

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