Multiple Instance Regression

Soumya Ray SRAY@CS.WISC.EDU
Department of Computer Sciences and Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, WI 53706

David Page PAGE@BIOSTAT.WISC.EDU
Department of Biostatistics and Medical Informatics and Department of Computer Sciences, University of Wisconsin, Madison, WI 53706

Abstract

This paper introduces multiple instance regression, a variant of multiple regression in which each data point may be described by more than one vector of values for the independent variables. The goals of this work are to (1) understand the computational complexity of the multiple instance regression task and (2) develop an efficient algorithm that is superior to ordinary multiple regression when applied to multiple instance data sets.

1. Introduction

The multiple instance problem (Dietterich et al., 1997) arises when the classification of every data point is not known uniquely. For instance, we might know that one of attribute vectors X_1 or X_2, or both, are responsible for a data point being classified as belonging to a certain class, but we may be unable to pinpoint which vector. This is frequently the case. For example, in drug design, we wish to distinguish molecules effective as drugs from ineffective ones. Here, training examples are in the form of *conformations* (3D structures) of a molecule, along with its class (active/inactive). However, a molecule may exist in a dynamic equilibrium of several conformations. While the observed activity will be a function of one or more of these conformations, it is typically impossible to determine which one(s). On the other hand, it is almost never the case that *all* conformations contribute to the observed activity. Hence it is desirable to learn a classifier which can take the multiple instance nature of these examples into account. Multiple instance problems arise in a variety of other domains as well, ranging from in-vitro fertilization (Saith et al., 1997) to image analysis (Maron & Lozano-Pérez, 1998).

It is worthwhile to note that in several applications of the multiple instance problem, the actual predictions desired are real valued. The drug design example is a case in point. While it is beneficial to be able to predict the active or inactive classification, our experience is that drug developers often prefer to see predicted *activity levels* of these molecules, expressed as real numbers. Most past research on the multiple instance problem has focused on the design of discrete classifiers. We investigate instead the task of learning to predict the value of a real valued dependent variable, under the assumptions of multiple regression, for data where the multiple instance problem is present. We call this task *multiple instance multiple regression*, or for brevity *multiple instance regression*.

Our investigation of multiple instance regression has two goals. The first is to understand the computational complexity inherent in the task of multiple instance regression—for example, we would like to know if a linear time algorithm exists as for ordinary regression. The second goal is to determine whether multiple instance regression has any advantage over ordinary regression when building classifiers for data sets where the multiple instance problem is present. In such cases, we could simply ignore the multiple instance problem, treat each instance as a distinct data point having the classification of the bag, and use ordinary regression. This is effectively the approach taken by Srinivasan and Camacho(1999) to incorporate linear regression literals into inductive logic programming (see Section 6). We wish to understand if multiple instance regression confers any benefit over this baseline method.

2. Task Definition

We define the task under consideration as follows. We are given a set of n **bags**. The i^{th} bag consists of m_i **instances** and a real valued class label y_i. Instance j

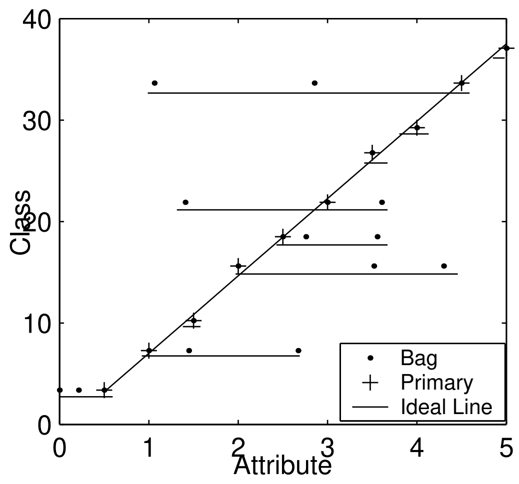

Figure 1. An example of a synthetic multiple instance regression problem in two dimensions. Each bag is underlined, and consists of at most three instances with different values for the real-valued attribute and sharing common real-valued class labels. The primary instances of each bag are shown as "+" symbols. The line that we would like to extract as the model for this data is also shown.

of bag i is described by a real valued attribute vector $\vec{X}_{ij}$ of dimension d. An example of a synthetic multiple regression problem is shown in figure 1. In the drug design example, each bag would be a molecule, and each instance a conformation of the molecule represented by an attribute vector.

We assume that the hypothesis underlying the data is a linear model with Gaussian noise on the value of the dependent variable (which is also the real valued label). Further, we assume that it is sufficient to model one instance from each bag, i.e. that there is some *primary* instance which is responsible for the real valued label. We limit the present paper to linear hypotheses for two reasons. First, multiple linear regression is probably the single most well-known and widely-used method of real-valued prediction. Second, multiple regression appears well-suited for the particular task of drug activity prediction(Hansch et al., 1962; Debnath et al., 1991) that was the original motivation for multiple instance learning. A linear hypothesis is intuitively plausible as a predictor for activity levels. It is natural to expect that activity levels will decrease exponentially as 3-dimensional distances between atoms in a molecule vary from the ideal distances. However, activity levels are typically recorded on a logarithmic scale, so the dependence between these and distances may be linear.

Ideally, we would like to find a hyperplane $\mathbf{Y} = \mathbf{X}\mathbf{b}$ such that

$$\mathbf{b} = \arg\min_{\mathbf{b}} \sum_{i=1}^{n} L(y_i, \vec{X}_{ip}, \mathbf{b}) \qquad (1)$$

where $\vec{X}_{ip}$ describes the primary instance of bag i, and L is some error function measuring the goodness of the hyperplane with respect to each instance. Intuitively, this describes the model as the best hyperplane in $\mathbb{R}^{d+1}$ with respect to the "correct" (primary) instances. However, the primary instances are unknown at training time, so this is impossible in practice. Nevertheless, we make the following informal conjecture.

Conjecture 1 *In most situations, a good approximation to the ideal can be obtained from the "best fit" hyperplane defined by*

$$\mathbf{b} = \arg\min_{\mathbf{b}} \sum_{i=1}^{n} \min_{j} L(y_i, \vec{X}_{ij}, \mathbf{b}), 1 \leq j \leq m_i$$

for large enough n.

For Conjecture 1 to be true, it is necessary that the non-primary instances in each bag are not a better fit to a hyperplane than the primary instances. A future direction of this work is to ascertain the conditions under which this conjecture is valid. Note that it is possible that the provided values may be noisy, so that the minimum L-error in conjecture 1 is not necessarily zero.For our algorithm, we use

$$L(y_i, \vec{X}_{ij}, \mathbf{b}) = (y_i - \vec{X}_{ij}\mathbf{b})^2$$

as is used for multiple regression.

It is clear that if $n < d+1$ (the dimension of the space) there are infinitely many hyperplanes with zero L-error with respect to a set of instances containing one instance from each bag, and the problem is trivial since any of these planes solves the constraint in conjecture 1. On the other hand, if $n \geq d+1$ a brute force approach trying all possible hyperplanes is exponential in m_i and n. In fact, the problem of minimizing the L-error for $n \geq d+1$ is intractable unless $P = NP$. We state this result in the following theorem.

Theorem 2 *The decision problem: Is there a hyperplane which perfectly fits one instance from each bag? is NP-complete for arbitrary n, d ($n \geq d+1$) and m_i at most 3.*

The proof proceeds by a reduction from 3SAT, and has been omitted for brevity. It is clear that the NP-completeness of the above decision problem implies the NP-hardness of the related decision problem: *Is there*

Multiple Instance Regression algorithm
Input: An integer R and n bags,where bag i is $\vec{X}_{i1}, \vec{X}_{i2}, \ldots, \vec{X}_{i,m_i}$; $\vec{X}_{ij}$ an attribute vector of dimension d.
Output: A hyperplane $\mathbf{Y} = \mathbf{Xb}$.

1. **Let** $GlobalErr = MAXDOUBLE$ (the maximum representable double precision value)
2. **For** $r = 1 \ldots R$
3.   Choose a random initial hyperplane $\mathbf{b}$ in $\mathbb{R}^{d+1}$.
4.   **Let** $BestErr = MAXDOUBLE$
5.   **Let** $ErrThisIter = 0$
6.   **Let** $Done = false$
7.   **Repeat**
8.   **Let** $I = \phi$
9.   **For** every bag $i = 1 \ldots n$ /* find new instances */
10.   **For** every instance $j = 1 \ldots m_i$
11.   Calculate the error of the instance with respect to the hyperplane: $L(y_i, \vec{X}_{ij}, \mathbf{b}) = (y_i - \vec{X}_{ij}\mathbf{b})^2$
12.   end **For** /* instances */
13.   **Let** $I = I \cup \{$the instance with the lowest error$\}$. Let this error be L_{min}.
14.   **Let** $ErrThisIter = ErrThisIter + L_{min}$
15.   end **For** /* bags */
16.   **if** $ErrThisIter \geq BestErr$ /* check convergence */
17.   **then** $Done = true$
18.   **else**
19.   **Let** $BestErr = ErrThisIter$
20.   **Let** $\mathbf{b}' = \mathbf{b}$
21.   Perform multiple regression over I to obtain a new hyperplane $\mathbf{b}$.
22.   **endif**
23.   **Until** $Done$
24.   Let the error of $\mathbf{b}'$ be E_{min}.
25.   **if** $E_{min} < GlobalErr$
26. $GlobalErr = E_{min}$
27. $\mathbf{b}'' = \mathbf{b}$
28.   **endif**
29. end **For** /* random restarts */
30. Return the plane $\mathbf{b}''$.

Figure 2. Multiple Instance Regression algorithm

a hyperplane which fits one instance from each bag such that the total L-error is $\leq e$? for some given positive constant e. This in turn shows that the general formulation of the multiple instance regression problem is NP-hard. Hence, we devise an approximation algorithm to solve our problem.

3. Algorithm

Analogous to approaches to other multiple instance learning tasks (Dietterich et al., 1997; Jain et al., 1994), we employ an Expectation Maximization (EM) algorithm, shown in figure 2. We start with an initial random guess at the hypothesis which is iteratively refined. Each iteration consists of two main steps. In the E step, we select an instance from each bag which has least L-error with respect to our current best guess at the correct hypothesis (hyperplane). In the M step, we refine our current guess of the hypothesis by using multiple regression to construct a new hyperplane from the set of instances selected in the previous step. These steps are repeated until the algorithm converges.

We provide an intuitive sketch of the proof of convergence. Note that a set of instances selected in the E step uniquely defines a hyperplane (step 21). Suppose at a certain step we have a set of instances I_k which has an L-error e_k with respect to our current guess at the hypothesis. In the next iteration, $I_{k+1} \neq I_k$ and $e_{k+1} < e_k$ (step 16). Since the error decreases monotonically, the set of instances can never repeat. However, there are only finitely many sets of instances that the algorithm can explore. Hence it must terminate in a finite number of steps.

EM algorithms are not deterministic, because the result of any run is influenced by the initial random starting point—in our algorithm, the starting hyperplane. Hence it is common to run an EM algorithm several times on any given data set, using "random restarts." The quantity R in the algorithm is the number of random restarts to be used. We have used an R of 10 in our experiments.

The algorithm is modular. We could choose any L-measure we wish (subject to convergence requirements as discussed above), and also any class of (possibly nonlinear) hypotheses to explore in step 21. We might, for instance, use an artificial neural network, as in a related approach taken by Jain et al. (1994), discussed in section 5.

4. Experiments

We have tested the algorithm thoroughly using synthetic data sets, comparing it with ordinary multiple regression and generating learning curves. This section describes the experimental setup and synthetic data experiments.

4.1 Experimental Setup

We generated synthetic data sets by choosing random hyperplanes. The generating program took as input an interval $\{x_{min}, x_{max}\}$, the dimension of an attribute vector d, the number of bags n and the maximum number of instances per bag m. For each bag, a random number of instances between $\frac{m}{2}$ and m were generated. For the first instance of a bag, independent co-ordinates were generated by moving in increments of $\frac{x_{max}-x_{min}}{n}$ from x_{min} to x_{max} along all dimensions. The y co-ordinate (the real-valued class label) was computed from the known hyperplane and Gaussian noise was added to it. The independent X co-ordinates of the remaining (non-primary) instances of each bag were drawn randomly according to two different distributions. In our first experiments, these were drawn according to a uniform distribution over $\{x_{min}, x_{max}\}$. To simulate cases where the X co-ordinates of different instances of the same bag are correlated, as might occur in the case of drug activity prediction, we also performed experiments using a Gaussian distribution in place of the uniform distribution. Here, each X co-ordinate of a non-primary point was drawn from a Gaussian whose mean was the value of that X co-ordinate from the primary instance and whose standard deviation was 10.0. Note that the non-primary instances share the y co-ordinate (class label) of the first instance.

In the learning curve experiments described below, we used a maximum of 10 instances per bag. The attribute vector describing each instance was a 20 dimensional real valued vector. x_{min} was set to 0 and x_{max} was set to 100. The distribution governing the Gaussian noise added to y was $N(0, 5)$. We generated the data using ten random hyperplanes in $\mathbb{R}^{21}$. We constructed six data sets using each hyperplane, containing 100 to 2500 bags. For each hyperplane, we generated test sets containing 1000 bags. Due to the paucity of time, we were unable to complete experiments with 2500 bags for the Gaussian distribution.

To evaluate our algorithm in these experiments, we generated test sets according to the same models as the training sets. We tested the algorithm using two measures of goodness. The first, which we shall call the *accuracy* measure, computes the fraction of primary instances that are among the set of instances closest to a given hyperplane. The higher this measure is, the better is our approximation to the ideal (1). The second measure is a *test set r-square* measure defined as follows:

$$R^2 = 1 - \frac{\sum_i (y_i - y_i^p)^2}{\sum_i (y_i - \bar{y})^2} \tag{2}$$

where y_i is the actual y value for the i^{th} bag, y_i^p is the predicted y-value for the (primary instance of the) i^{th} bag, and $\bar{y}$ is the mean y value over the training and test set. This measure therefore computes the improvement in fit of our plane over the simple plane $y = \bar{y}$. If measured on the training set with respect to the set of points closest to the hyperplane, this measure is the usual R^2 measure and is positively correlated with our approximation to the "best fit" hyperplane (conjecture 1). We note that in our algorithm, we try to explicitly optimize the training set R^2 measure in this way(step 21). The accuracy measure is optimized contingent on the truth of our assumption that the best fit line is a good approximation to the ideal.

Since we tested on synthetic data, it was quite simple to compute these measures for any hyperplane. We generate data so that the first instances of any bag are the primary instances. After a hyperplane is generated by our algorithm, we can compute its accuracy over a data set by computing the fraction of points closest to it that were also the first instances in each bag. We can also directly compute the value of R^2 for our approximation to the ideal by choosing the y_i in (2) from the primary instances of each bag.

In all figures that follow, "MIP" represents our algorithm, "Base" represents ordinary regression and "Best" represents regression over the primary points.

4.2 Learning Curves

We constructed the learning curves in Figures 3 and 4 to test our primary hypothesis: as we get more data points (bags), with all other variables held constant, the hyperplanes produced by the algorithm should

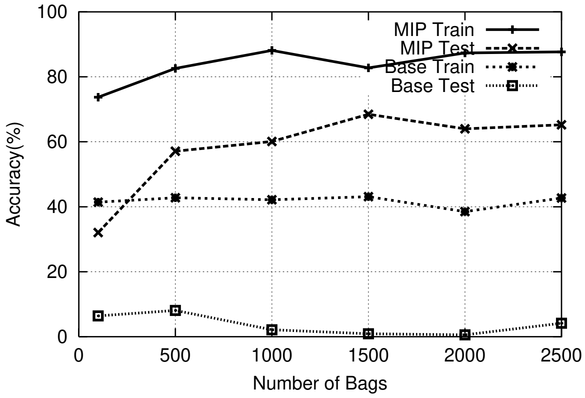

Figure 3. Accuracy Learning Curves with Uniform Distribution

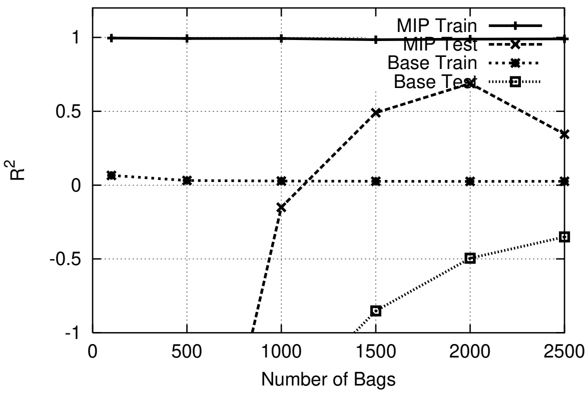

Figure 4. R^2 Learning Curves with Uniform Distribution

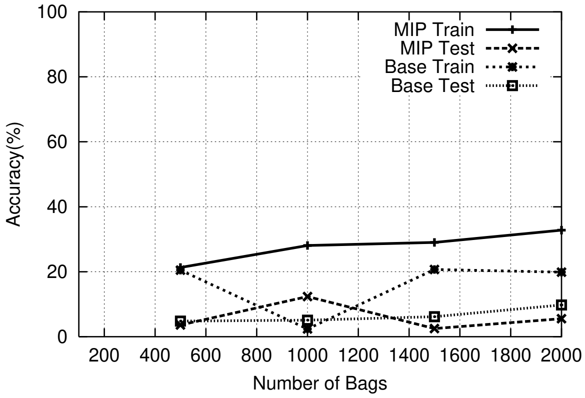

Figure 5. Accuracy Learning Curves with Gaussian Distribution

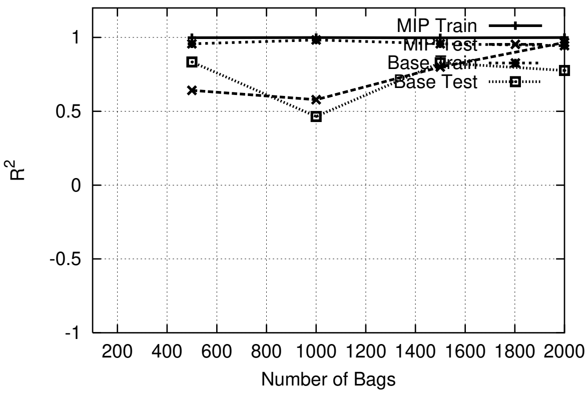

Figure 6. R^2 Learning Curves with Gaussian Distribution

converge towards the ideal, as measured by the accuracy and R^2 measures. We compare a baseline algorithm with our algorithm. The baseline algorithm performs simple multiple regression over the entire set of data points ignoring the multiple instance aspect of the problem.

The results in Figures 3 and 4 clearly indicate that multiple instance regression confers benefit over ordinary regression. For some indication of significance, the difference in test accuracies at 1000 bags is significant to a level of 10^{-15} according to the standard sign test. Nevertheless, these results raise a number of questions which we next seek to answer.

First, it seems likely that the significant benefit of multiple instance regression over ordinary regression arises in part because the values of the independent variables of the non-primary instances in a bag are completely uncorrelated with the values of those variables in the primary instance of the bag. This independence may not always be the case in practice. To test the contribution of this independence, we choose a way of introducing a high degree of correlation into our synthetic data. We repeat the same experimental setting but with the value of an independent variable in a non-primary instance chosen according to a Gaussian whose mean is the value of that variable in the primary instance. This perhaps induces a more extreme correlation than would be expected in practice. Figures 5 and 6 show that in this case ordinary regression performs nearly as well as multiple instance regression. Another observation is that it seems easier to obtain higher R^2 values for this setup. A possible reason for these observations is suggested by figure 7. Here, we compute an R^2 measure that estimates the amount of linearity in the non-primary points alone[1], when the Gaussian distribution is used to generate them. In this measure we let y_i^p in equation (2) be the prediction of our algorithm, while $\bar{y}$ is replaced by the prediction of a plane obtained by regression over the non-primary points alone. In figure 7, we plot this modified measure against the number of dimensions and the number of instances. We note that, by modifying the distribution, we have introduced a significant amount of "random linearity". This contributes to the poor performance of the algorithm, since it is much more likely

[1]We thank the anonymous referees for this suggestion.

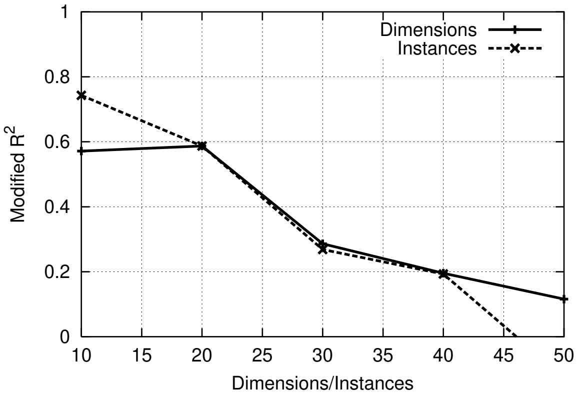

Figure 7. Modified R^2 vs. Dimensions and Instances

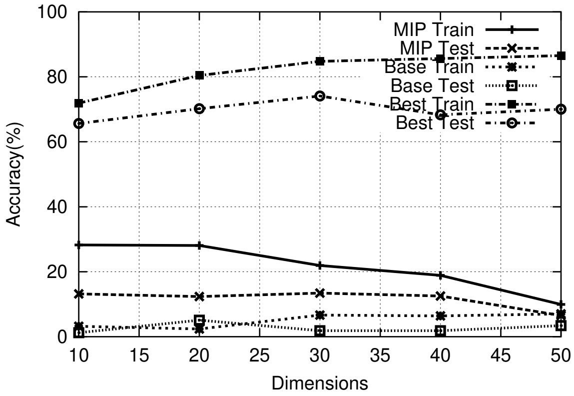

Figure 8. Accuracy vs. Dimensions Curves

to be trapped in a local minimum of the error measure.

4.3 Variation in Dimensions and Instances

We next study the variation in the performance of multiple instance regression with the number of dimensions (independent variables) and number of instances per bag. Figures 8 and 9 plot accuracy and R^2, respectively, against number of dimensions, while Figures 10 and 11 plot accuracy and R^2 against number of instances per bag. We use the Gaussian distribution in generating the synthetic data for these experiments. The number of bags is held constant at 1000. As expected, the accuracy decreases with both increasing dimensions and instances. However, the decrease is much more rapid with increasing number of instances. This is also expected, since the combinatorial factor in the algorithm's search arises primarily from the number of instances in a group. Hence, as the number of instances increases, we should use more random restarts to enable the algorithm to find a good solution.

In all of the experiments with the Gaussian distribution, we note that training set R^2 (the measure which we actually optimize) is very close to 1.0. However, this does not necessarily result in good test set accuracy. A further verification is provided by figure 7, which indicates increasing linearity in the non-primary points with increasing dimensions and instances. We have plotted the accuracy and R^2 measures for the ideal plane in figures 8 to 11 as "Best". From these results, we see that it is possible to achieve high accuracy and R^2 simultaneously. We may reasonably conclude that, when non-primary instances are generated according to a distribution more complex than the uniform, merely optimizing the R^2 measure will not be enough. Therefore, there is much room for improvement in the algorithm.

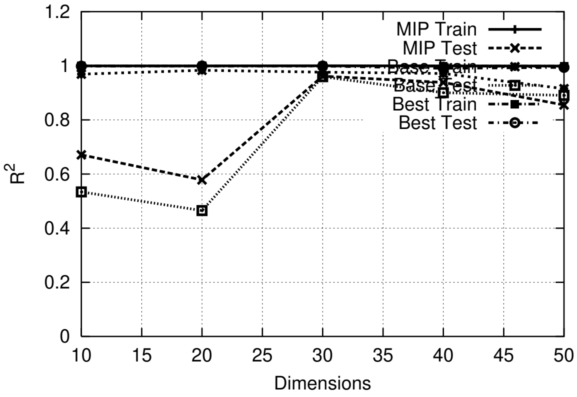

Figure 9. R^2 vs. Dimensions Curves

4.4 Runtime Complexity

The multiple instance algorithm (figure 2) has two main loops. The outer loop (step 2 to step 29) runs R times, which is a constant. The inner loop (step 7 to step 23) runs an undetermined, finite number of times. In each inner cycle, we compute the error for every instance and do multiple regression over the best set found. This takes $\Theta(d \cdot m \cdot n)$, where $m = \max_i(m_i)$, and is independent of the specific hyperplane being looked at. If the parameters d and m are held constant, the inner loop is $\Theta(n)$. Hence, the quantity of interest is the average number of cycles taken by the algorithm to converge, because this determines the runtime complexity as the parameters n, m and d change. We plot this quantity against the parameters in figures 12 and 13. It is interesting to note that the number of cycles appears to increase linearly or sub-linearly with the number of bags (with the increase being nearly linear for Gaussian), but number of cycles relative to dimension or instances appears bounded by a constant. We note that the algorithm's search is carried out over the space of combinations of instances, so that increasing the number of instances impacts runtime more than increasing dimensions. On the other hand, appeal-

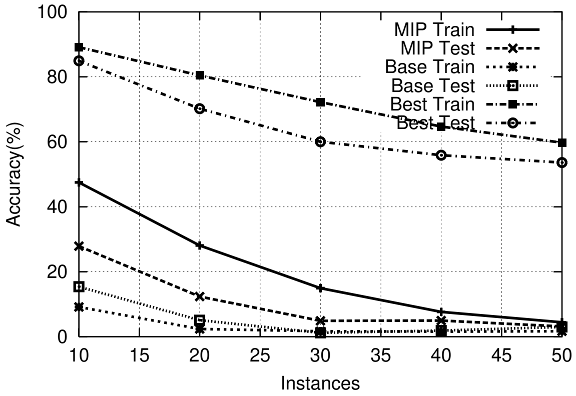

Figure 10. Accuracy vs. Instances Curves

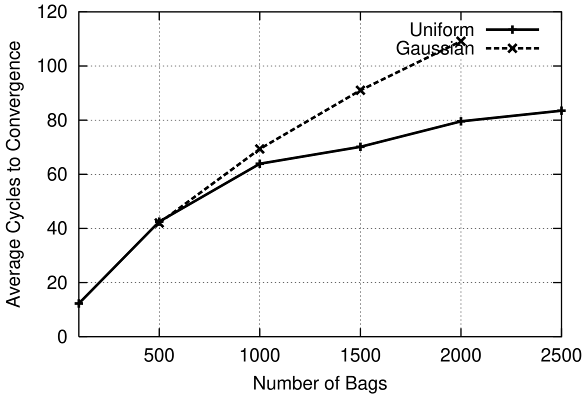

Figure 12. Average number of cycles with increasing n

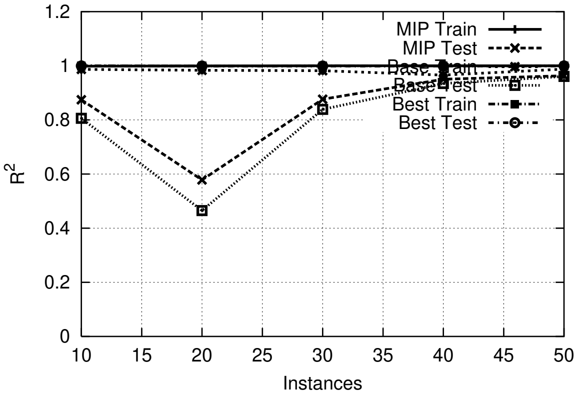

Figure 11. R^2 vs. Instances Curves

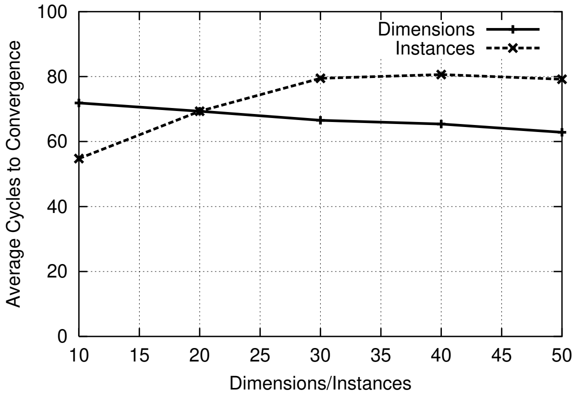

Figure 13. Average number of cycles with increasing dimension and instances (Gaussian distribution only)

ing to figure 10, it is more likely that as instances increase, the algorithm is finding spurious planes. This is possibly why the average number of cycles to convergence does not show much increase as the number of instances increases.

5. Related Work

There are several existing approaches to handling the multiple instance problem for discrete classification. Dietterich et al. (1997) describe algorithms which are applicable when the classifiers are axis-parallel rectangles. Here, the axes are features. Each example (bag) can be described by a range of values on each axis corresponding to the minimum and maximum values for that feature among the instances constituting the bag. For example, in the drug activity prediction task, these features might be distances to the molecule surface from some chosen origin. These algorithms learn a rectangle in this feature space that covers the most positive examples (at least one instance from each) while not covering negative examples. Maron et al. (1998) describe a general approach called Diverse Density for multiple classification. Diverse Density uses a probabilistic approach to maximize a measure of the intersection of the positive bags minus the union of the negative bags in feature space. The algorithms of both Dietterich et al. and Maron et al. were tested on a problem similar to drug activity prediction, that of classifying *musk* and non-*musk* molecules. Unfortunately, while there are degrees of musk odor, there is no data set available with real-valued measurements of musk odor for use with multiple instance regression.

Predicting continuous quantities in the presence of the multiple instance problem has received less attention. The most closely related work to ours is by Jain and colleagues (1994). They designed a system called COMPASS for drug activity prediction, which uses an EM approach combined with a neural network. This system returns real valued estimates of the activity of a candidate molecule. COMPASS is specific to the domain of drug activity prediction, and its expectation step involves computing alternative low-energy conformers for the molecules (in selected iterations) and re-aligning the chosen conformers of the molecules with one another. Nevertheless, it should be possible to substitute a more general purpose E step to change

COMPASS into a general-purpose algorithm for multiple instance prediction.

6. Conclusion and Future Work

In this work, we have introduced the task of multiple instance regression and noted that, whereas ordinary regression admits a linear-time algorithm, multiple instance regression is NP-hard. We have therefore presented an EM algorithm for multiple instance regression that is not specific to any domain, and because of its modularity may be extended to more complex models. We have shown using synthetic data that, when the non-primary instances are not correlated with the primary instances, it significantly outperforms ordinary regression. Furthermore, the number of cycles required appears to grow at most linearly with the number of bags, and does not appear to grow beyond a constant bound with the number of dimensions and number of instances per bag.

Nevertheless, when non-primary instances are highly correlated with primary instances the algorithm does not significantly outperform ordinary regression. Furthermore, the algorithm still falls short of the results of regression given the primary instances only (the ideal). These observations suggest there is room for much further research into multiple instance regression. One immediate direction is to try alternative EM algorithms that do not select a single instance of each bag, but instead weight each instance by likelihood. An important theoretical direction for this work is to determine when Conjecture 1 is applicable.

Perhaps the most important area for immediate further work is in the application to real-world data sets. This would provide insight, for example, into the degree of correlation one can expect between primary and non-primary instances. We currently are constructing such sets within the context of combining multiple instance regression with inductive logic programming (ILP). We close with a description of this work.

In drug design and several other domains, the value of one of the variables in a clause (e.g., *Activity*) might be assumed to be a linear function, with Gaussian noise, of other variables in the clause. Let us call the first variable the "dependent variable" and the others the "independent variables." In such cases, it is natural to combine ILP with linear regression. Indeed, Srinivasan and Camacho (1999) first made this assumption and applied their approach to physical modeling (pole-and-cart problem) and prediction of mutagenicity. A difficulty in this approach is that the variables in a clause may take *multiple bindings*, and one does not know which bindings are responsible for the value of the independent variable. In the work of Srinivasan and Camacho, each different vector of bindings for the independent variables gave rise to a distinct data point for regression. This is analogous to using the "Base" algorithm of our paper. But for many applications, all we know is that at least one of the vectors of bindings is responsible for the value of the dependent variable, rather than all. Hence it is natural to model the regression task as a multiple instance problem. We are currently developing and obtaining public data sets of compounds with continuous activity levels for experiments with the combination of ILP and multiple instance regression.

Acknowledgements

The authors wish to thank Olvi Mangasarian for helpful comments and the use of computational resources. This material is based upon work supported by the National Science Foundation under Grant No. 9987841.

References

Debnath, A., de Compadre, R. L., Debnath, G., Schusterman, A., & Hansch, C. (1991). Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, *34*, 786 – 797.

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

Hansch, C., Maloney, P., Fujita, T., & Muir, M. (1962). Correlation of biological activity of phenoxyacetic acids with Hammett substituent constants and partition coefficients. *Nature*, *194*, 178–180.

Jain, A. N., Koile, K., & Chapman, D. (1994). Compass: Predicting biological activities from molecular surface properties. Performance comparisons on a steroid benchmark. *Journal of Medicinal Chemistry*, *37*, 2315–2327.

Maron, O. (1998). *Learning from ambiguity*. Doctoral dissertation, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

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

Saith, R., Srinivasan, A., Michie, D., & Sargent, I. (1997). The relationship between embryo, oocyte and follicular features and the developmental potential of human IVF embryos. *Human Reproduction (Submitted)*.

Srinivasan, A., & Camacho, R. C. (1999). Numerical reasoning in an ILP system capable of lazy evaluation and customized search. *Journal of Logic Programming*, *40*, 185–214.