Abstract

Most prior work on multiple-instance learning has been for concept learning, yet for drug activity prediction, the label is a real-valued affinity measurement giving the binding strength. We present extensions of $k$-NN, Citation-$k$NN, and the diverse density algorithm for the real-valued setting and study their performance on boolean and real-valued data. We also provide a method for generating chemically realistic artificial data.

1. Introduction

The multiple-instance learning model is becoming increasingly important. Unlike standard supervised learning in which each instance is labeled in the training data, here each example is a set (or bag) of instances which receives a single label equal to the maximum label among the instances in the bag. The individual instances are not given a label. The goal of the learner is to generate a hypothesis to accurately predict the label of previously unseen bags.

The multiple-instance model was motivated by the drug activity prediction problem where each instance is a possible configuration (or shape) for a molecule of interest and each bag (example) contains all low-energy (and hence likely) configurations for the molecule (Dietterich, Lathrop, & Lozano-Pérez, 1997). There has been a significant amount of research directed towards this problem (Auer 1997; Maron & Lozano-Pérez, 1998, Maron 1998, Wang and Zucker 2000). Other applications for the multiple-instance model have also been proposed (Maron & Raton, 1998; Ruffo 2000).

Prior research performed under the multiple-instance model is for concept learning (i.e. boolean labels). Binding affinity between molecules and receptors is quantitative, borne out in quantities such as the energy released by the molecule-receptor pair upon binding and hence a real-valued label of binding strength is preferable. The only real data sets available as benchmarks are the Musk1 and Musk2 data sets provided by Dietterich et al. (1997). Dietterich et al. say “The only aspect of the musk problem that is substantially different from typical pharmaceutical problems is that the musk strength is measured qualitatively by expert human judges, whereas drug activity binding is usually measured quantitatively through biochemical assays.” We argue later that other aspects of the musk data sets are also atypical.

While our paper focuses on studying the real-valued multiple-instance setting, we also want to deepen our understanding of algorithms that have been proposed for the boolean setting. Along with some real-valued benchmarks, additional boolean benchmarks are needed. Wang and Zucker (2000) say, “Although the two adaptation algorithms of $k$NN performed remarkably well, the basic reasons why they acquired such high accuracy on the musk data sets are unclear.” The ability to generate artificial data sets will help answer such questions.

In this paper, we study extensions of the diversity density, $k$-NN, and citation-$k$NN algorithms for the real-valued multiple-instance setting. We look at both the prediction error (where all labels $> .5$ are treated as 1 and the rest are treated as 0) and the squared loss$^1$. We provide two baselines with which to compare our work. In the first baseline, a random bag from the training data is selected and its label is returned. As a second baseline, we use the standard unweighted $k$-NN algorithm by converting the multiple-instance problem to a standard supervised learning problem by assigning each point the label of its bag. Then, the standard (single-instance) $k$-NN algorithm using the Euclidean distance is used. We compare the performance of these algorithms when using real-valued data to their performance when using boolean labels obtained by rounding

---

$^1$We assume without loss of generality that the labels are in $[0, 1]$. 

the real-valued labels to 0 or 1. Even when the training data has only labels of 0 or 1, the prediction is real-valued. For example, for \( k \)-NN, the average label from the \( k \) nearest neighbors is output. We report the squared loss between the underlying real-valued label and the real-valued prediction made by the algorithm that received only boolean labels. Not surprisingly, we found the squared-loss is greatly reduced when using real-valued labels versus the boolean labels.

Our empirical work has provided some valuable new insights about these algorithms. A secondary contribution of our work is a procedure for generating chemically realistic artificial data in which you can control factors such as the number of conformers, the number of relevant features, and their degree of relevance. We have placed the data sets used in this paper at www.cs.wustl.edu/~sg/multi-inst-data/.

2. Previous Work

In their seminal paper, Dietterich et al. (1997), presented three methods for learning axis-aligned boxes (often referred to as APR for axis-parallel rectangles) in the multiple-instance model. They presented the “inside-out” algorithm which starts with a point in the feature space and “grows” a box with the goal of finding the smallest box that covers at least one example from each positive bag and no examples from any negative bag. Then they expand the resulting box (via a statistical technique). When appropriately tuned, their algorithm gives 89% accuracy on Musk2.

The work of Dietterich et al. was preceded by the work of Jain et al. (1994) in which they presented COMPASS which as an APR-like neural network algorithm which is robust to errors in the initial alignment of the molecules. While COMPASS can handle real-valued labels, we are not aware of any reported results on any available real-valued data sets.

Auer (1997) presented an algorithm that learns using simple statistics and hence avoids some potentially hard computational problems that were required by the heuristics used by Dietterich et al. Their algorithm worked quite well on the Musk2 data set (obtaining a 84% accuracy) despite the fact that they assumed each point in a bag was drawn independently of the others.

Maron and Lozano-Pérez (1998) described a framework called Diverse Density (see also Maron, 1998). When describing the shape of a molecule by \( n \) features, one can view each configuration as a point in a \( n \)-dimensional feature space. As the molecule changes its shape, it traces out a manifold through this \( n \)-dimensional space. The diverse density at a point \( p \) in the feature space is a measure of both how many different positive bags have an example near \( p \), and how far the negative instances are from \( p \). They use gradient ascent with multiple starting points (namely, starting from each point from a positive bag) to find the point that maximizes the diverse density. Their algorithm obtained 82.5% accuracy on the Musk2 data.

More recently, Wang and Zucker (2000) proposed a lazy learning approach to multiple-instance learning by applying a variant of the \( k \) nearest neighbor algorithm (\( k \)-NN). To compute the distance between bags \( b_1 \) and \( b_2 \) they used the minimum distance between a point in \( b_1 \) and a point in \( b_2 \). While a standard \( k \)-NN approach did not work well, by also using citers of \( p \) (points who include \( p \) as one of its nearest-neighbors) as well as \( p \)'s nearest neighbors they reached a 92.4% accuracy on Musk1 and 86.3% accuracy on Musk2.

3. Artificial Data Generation

We first review how the musk data sets were generated. Dietterich et al. (1997) constructed a molecular surface for each conformation by orienting all molecules with respect to a common origin, and then using 162 uniformly distributed random sampling rays radiated from the origin. The length of the molecular surface along each ray was recorded as a feature value. These 162 features were supplemented to obtain a 166-dimensional feature vector. Boolean classifications were obtained by culling molecules strongly believed (by a human expert) to be musk as positive examples and strongly believed to be non-musk as negative examples — no “borderline” data was included.

The exclusion of borderline data is one of several factors which make the musk data sets easier than one would typically expect. Based on the structural requirements for nitro-free aromatic musk molecules [Fehr, 1989], the interaction between a musk molecule and the receptor can be approximately categorized to three different interactions. Accordingly, we would expect the degree of importance (or relevance) for the relevant variables to be one of three values depending on which kind of interaction occurs. Furthermore, musk molecules have a closely packed structure and the binding involves large portions of the molecule. Since the rays used to represent the shape of the molecule were approximately uniformly emanated from the origin, a considerable number of rays will pass through the structural motifs of the molecule that involve binding. Hence a large number of features will be relevant.

Because of these aspects of the musk data sets, some algorithms may work very well on Musk1 and Musk2 but not work as well on more typical data. The artificial data provides some additional benchmarks
with much more varied characteristics. Furthermore, knowledge about the parameters used to generate the artificial data enables us to develop a better understanding of the strengths and limitations of various learning algorithms. For example, we believe that if we knew important factors such as which features were relevant, then we could obtain substantially better performance. However, we had no way to test this hypothesis on the real data since we do not know which features are relevant. While some of our findings are as expected, others have been surprising. Also with the artificial data we can vary, in a controlled way, parameters such as the number of relevant variables and the number of relevance levels and see how various algorithms’ performance is affected by these changes.

We created our artificial data by generating an “artificial receptor.” “Artificial molecules” were then generated with each feature value considered as the distance from the origin to the molecular surface when all molecules are in the same orientation. Each feature has a scale factor to represent the degree of importance of that feature in the binding process, with 1 as the most relevant and 0 as irrelevant. Artificial molecules (bags) with 3 to 5 instances per bag were generated for each artificial receptor. Let \( \{B_i, t_i\} \) denote the \( i^{th} \) labeled bag in data set \( B \), \( B_{ij} \) denote the \( j^{th} \) instance of bag \( i \), and \( B_{ijk} \) denote the feature value of instance \( B_{ij} \) on feature \( k \). Given the artificial receptor (target) \( t \), let \( t_k \) represent the value of \( t \) on feature \( k \), and let \( s_k \) be the scale factor for feature \( k \).

The binding energies between the artificial molecules and receptor were calculated on the basis of a widely used empirical potential for intermolecular interactions, the Lennard-Jones potential (Berry, 1980)

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

where \( \epsilon \) is the depth of the potential well, \( \sigma \) is the distance at which \( V(r) = 0 \), and \( r \) is the internuclear distance. We choose the Lennard-Jones (LJ) model because of its mathematical simplicity and ability to qualitatively mimic the real interaction between molecules. In generating the artificial data, we assume that \( \sigma \) and \( \epsilon \) are known. The binding energy of molecule \( B_i \) to \( t \) is

\[
E_{B_i} = \max_{B_{ij} \in B_i} \left\{ \sum_{k=1}^{n} s_k \cdot V(t_k - B_{ijk}) \right\}
\]

where \( n \) is the number of features.

The label for molecule \( B_i \) is the ratio of \( E_{B_i} \) to the maximum possible binding energy \( E_{max} \) possible:

\[
\text{Lab}_{B_i} = E_{B_i} / E_{max}
\]

where \( E_{max} = -\epsilon \cdot \sum_{k=1}^{n} s_k \). In the real data we have the labels were not uniformly distributed. Accordingly, during the generation of artificial molecules, the feature values of one instance in a molecule were generated in a controlled manner to mimic this behavior.

As a naming convention we use \( \text{LJ-r}_{.f.} \) where \( r \) is the number of relevant features, \( f \) is the number of features (i.e., dimensions), and \( s \) is the number of different scale factors used for the relevant features. If \( s = 2 \) then half of the relevant scale factors were 0.5 and the other half were 1.0. When \( s = 4 \) then 1/4 of the relevant scale factors are set to each of 0.25, 0.5, 0.75, 1.0, and so on. As a default a bag with label 1.0 is generated by setting each of the relevant features to the correct value and then randomly picking a value for the irrelevant features. We also considered several variations. For some data sets the maximum label was 0.9 (indicated by the “-0.9” suffix). Finally, to partially mimic the musk data in some data sets we only used labels that were not near 1/2 (indicated by the “S” suffix) and all scale factors for the relevant features were randomly selected between [0.9, 1].

We have obtained one real data set\(^3\) that has real-valued affinity values. This data set has 283 features and 139 bags with an average of 32.5 points per bag. Only 29 bags have labels that were high enough to be considered as “positive.” A summary of the key characteristics of our artificial data sets as well as the real data sets are given in Table 1.

<table>
<thead>
<tr>
<th>data set</th>
<th>number relevant</th>
<th>number features</th>
<th># scale factors</th>
<th>num. bags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affinity</td>
<td>-</td>
<td>283</td>
<td>-</td>
<td>139</td>
</tr>
<tr>
<td>Musk1</td>
<td>-</td>
<td>166</td>
<td>-</td>
<td>92</td>
</tr>
<tr>
<td>Musk2</td>
<td>-</td>
<td>166</td>
<td>-</td>
<td>102</td>
</tr>
<tr>
<td>L.J-r.283.s</td>
<td>r</td>
<td>283</td>
<td>s</td>
<td>200</td>
</tr>
<tr>
<td>L.J-r.166.1</td>
<td>r</td>
<td>166</td>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>L.J-r.30.s</td>
<td>r</td>
<td>30</td>
<td>s</td>
<td>60</td>
</tr>
</tbody>
</table>

\(^3\)Jonathan Greene from CombiChem provided us with the Affinity data set. However, due to the proprietary nature of it we cannot make it publicly available and do not have any information about how the features were generated.

4. Nearest Neighbor Based Approach

One set of approaches we evaluated are based on the nearest neighbor algorithm. In particular, we consider a multiple-instance variant of unweighted \( k \)-NN in which the distance between bags \( B_i \) and \( B_j \) are defined as the minimum Euclidean distance between a point in \( B_i \) and a point in \( B_j \). Wang & Zucker (2000) called this distance measure the minimal Hausdorff distance. The prediction made for bag \( B \) is the
average label of the $k$ closest bags\(^4\). The other approach we used is a variation of citation-kNN (Wang & Zucker, 2000). Given a bag $B$, the $C$ nearest citers of $B$ include bag $B_i$ if and only if $B$ is one of the $C$ nearest neighbors of $B_i$. Note that the number of $C$ nearest citers is generally not $C$. Citation-kNN makes a prediction for bag $B_i$ by taking the average value of the $R$ nearest neighbors and $C$ nearest citers. In all of our experiments, we let $R = 3$ and $C = 5$.

Since the computation of the distance between two points is calculated using all features, when there are many irrelevant features, the distance between two points can be dominated by the irrelevant features. One way to address this problem is to stretch the axes (shortening the axes corresponding to less relevant features and lengthening the axes corresponding to more relevant features). We refer to $s_1, \ldots, s_n$ as the scale (or relevance) factors where $s_i$ defines the relevance of feature $i$ in computing the target value. Using the artificial data, we compared the results obtained when using (1) the true scale factors (i.e., those used in generating the data) to rescale the axes, (2) when projecting out all features with a scale factor less than 1/2, and (3) when projecting out all features with a scale factor of 0. For both (2) and (3), there is no rescaling of the axes for the features that are used. Over a wide range of data sets, 2-NN, 8-NN, and citation-kNN perform best when using option (3) in which all relevant features are treated equally. We found this surprising since we had expected that option (1) would be best. Further studies are needed to explain this finding. In addition, this suggests that finding a way to estimate which features are relevant will lead to better results.

We have used a very simple heuristic based on the MULTINST algorithm (Auer, 1997) to estimate which features are relevant. To estimate whether dimension $d$ is relevant, we project the points onto dimension $d$ and apply a low-pass filter. If the resulting graph has exactly one peak, we estimate that this is a relevant dimension and otherwise we set the scale factor to 0. The results are very preliminary, but in some cases we obtained an improvement to using no scaling.

In Table 2 we show sample results obtained under these variations for citation-kNN. The particular group of data sets is one in which our estimation technique worked better than average. However, the variations between the three options which require knowledge of the true scale factors are representative of what we found across all data sets for citation-kNN, 2-NN, and 8-NN. When reporting our results we show those obtained when using option (3) and label it as “ideal.”

5. Diverse Density Based Approach

In this section we describe our extension of the diverse density algorithm of Maron and Lozano-Pérez for the real-valued setting. Intuitively, the diversity density of a point $t$ is just the likelihood (with respect to the data) that $t$ is the target. Let $B = \{ (B_1,t_1), \ldots, (B_k,t_k) \}$ be the training data. Let $B_{ij}$ denote the $j^{th}$ instance of bag $i$, and $B_{ijk}$ denote the feature value of instance $B_{ij}$ on feature $k$. The diverse density of possible target point $t$ is defined as $DD(t) = Pr(t | B) = Pr(B | t) Pr(t) / Pr(B)$. We assume uniform priors and so the goal is to search for a $t$ that maximizes $Pr(B | t)$. Assuming the points in $B$ are independent yields $Pr(B | t) = \prod_{i=1}^{k} Pr(B_i | t)$. By Bayes’ rule, $Pr(B_i | t) = Pr(t | B_i) Pr(B_i) / Pr(t)$. We assume an uniform prior on the targets and that $Pr(B_i)$ is constant with respect to $t$. Hence the goal is to maximize $\prod_{i=1}^{k} Pr(t | B_i)$. The key modification required in moving to the real-valued setting is in estimating $Pr(t | B_i)$. We let

$$Pr(t | B_i) = (1 - |t_i - Label(B_i | t)|) / Z$$

where $Label(B_i | t)$ is the label $B_i$ would receive for target $t$ and $Z$ is a normalization constant.

We consider two formulas for $Label(B_i | t)$. The first is that of Maron (1999) in which

$$Label(B_i | t) = \max_{j} \left\{ \exp \left( - \sum_{d=1}^{n} (s_d (B_{ijd} - t_d))^2 \right) \right\}$$

(2)

where the target is defined by feature values $t_1, \ldots, t_n$ and scale factors $s_1, \ldots, s_n$, and the softmax is used.
to approximate the maximum so that it can be differentiated. If each \( t_i \in \{0, 1\} \) and we use Equation (2), then our algorithm reduces to the standard diverse density algorithm. We also tried using the LJ formula by setting \( \text{Label}(B_i | t) = E_{B_i} / E_{\text{max}} \).

The point \( t \) that maximizes \( \prod_{i=1}^b Pr(t_i | B_i) \) is found using a gradient ascent search over the \( 2n \) dimensional space defined by \( t_1, \ldots, t_n, s_1, \ldots, s_n \) using as multiple starting points the features \( t_1, \ldots, t_n \) from each point in a bag with the maximum label and 0.1 for each \( s_i \).

6. Empirical Results

In this section we report on our results. In order to study the difference in performance when using real-valued data versus boolean data, for all of the data sets we ran experiments using both the given real-valued labels and a boolean label obtained by rounding the label to 0 or 1. For each of these runs, we report the prediction error (i.e. the number of prediction mistakes divided by the number of bags in the test set) where for real-valued labels and predictions we used 0.5 as the cutoff between positive and negative. We also report the squared loss. For the nearest neighbor algorithms we use leave-one-out cross validation and for diverse density we used a separate test set.\(^5\)

6.1 Nearest Neighbor Results

A summary of our results when using the nearest neighbor based approaches are shown in Table 3. We tested 2-NN, 8-NN, and citation-kNN (with \( R=3 \) and \( C=5 \)). 8-NN performed approximately 75\% better than 2-NN in terms of both the prediction error and squared loss and hence we do not report these results. The performance of 8-NN and citation-kNN are very similar which is very different than what is found when using the musk data sets. For Musk1, 8-NN has error 20.7\% whereas citation-kNN has error 10.9\%. Similarly, for Musk2, 8-NN has error 27.5\% whereas citation-kNN has error 14.7\%. Further studies are need to explain why the results for the musk data sets are different in this respect than for our artificial data sets.

In Figure 1 are the results obtained using Citation-kNN on the Affinity data with boolean labels (on the left) and with real-valued labels (on the right). Figure 2 are some results obtained using artificial data. In the Affinity set all of the negative labels are clustered near 0.4 and we believe this is part of why the performance is better. Further tests are needed to verify this theory.

\(^5\)We performed tests for the nearest neighbor algorithms using the test set and obtained comparable results to leave-one-out cross validation.

In comparing the results obtained when using boolean versus real-valued labels, as one would expect, the expected loss is much lower when real-valued labels are used. Interestingly, the number of classification errors is slightly less, in general, when using boolean labels. For the Affinity data set, when using boolean labels the prediction accuracy was 86.3\% and the average squared lost was 0.1094. In contrast, when using the real-valued labels, the prediction accuracy was 85.6\% and the average squared lost was 0.0124. This difference can be seen visually in Figure 1. So in terms of reducing squared loss there are tremendous benefits to using real-valued data.

6.2 Diverse Density Results

We compare the performance of citation-kNN with the diverse density algorithm when using Maron’s formula for computing the labels (Equation (2)) and with all scale factors initially 0.1. A summary of these results are in Tables 4 and 5. The diverse density algorithm only performs better for the small data sets (and in one of the larger boolean data sets). The most likely explanation is that with a smaller search space, the gradient ascent search is successful. It is important to note that the diverse density algorithm runs orders of magnitude slower than citation-kNN.

In other simulations results (not shown), we found some interesting phenomena. Since the LJ formula for computing the label of a point is much more chemically realistic than the Maron’s formula, we use it to generate our artificial data. One would expect that using the LJ formula within the diverse density algorithm would yield better results. However, when using the LJ formula with any starting scale factors other than the true ones, the search stopped after only a few steps whereas with the Maron formula (with all initial scale factors of 0.1) the search continued much longer and reached a better point. We believe that Maron formula is more robust in terms of searching when started from a “wrong” point. Consider the results shown in Figure 3 in which the Maron formula with initial scale factors of 0.1 is used. Interestingly, starting at a point that is not at the target point actually gives better results which is presumably because when all of the relevant features begin at their correct value, then the search stops after only a few rounds and the scale factors are not able to adjust. However, when the maximum label is 0.9 there is “more room” for the search procedure to work and hence the results are better.

Another important observation we made is that the final scale factors obtained by the diversity density algorithm have little relation to the true scale factors.
### Table 3. Overview of nearest neighbor results when using real-valued data. When using the random base the average prediction error was 46.1% and the average squared loss was $1.433$. The last line of the table shows the average prediction error and squared loss for when the labels are rounded to 0 or 1 (i.e. boolean labels). For the boolean setting, the random base had an average prediction error of 46.1% and an average squared loss of $1.3065$.

<table>
<thead>
<tr>
<th>data set</th>
<th>NN baseline</th>
<th>8-NN “ideal” no scaling</th>
<th>est. scaling</th>
<th>citation k-NN “ideal” no scaling</th>
<th>est. scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>LJ-160.166.1-S</td>
<td>30.4 .0705</td>
<td>0.0 .0028</td>
<td>0.0 .0032</td>
<td>0.0 .0063</td>
<td>0.0 .0013</td>
</tr>
<tr>
<td>LJ-160.166.1</td>
<td>35.8 .0477</td>
<td>5.4 .0027</td>
<td>5.4 .0030</td>
<td>5.4 .0040</td>
<td>4.3 .0011</td>
</tr>
<tr>
<td>LJ-80.166.1-S</td>
<td>41.3 .0952</td>
<td>0.0 .0038</td>
<td>0.0 .0042</td>
<td>0.0 .0104</td>
<td>0.0 .0025</td>
</tr>
<tr>
<td>LJ-80.166.1</td>
<td>32.6 .0479</td>
<td>6.5 .0063</td>
<td>10.8 .0890</td>
<td>9.7 .0079</td>
<td>1.0 .0037</td>
</tr>
<tr>
<td>LJ-150.283.2</td>
<td>7.5 .0480</td>
<td>0.0 .0023</td>
<td>27.5 .0709</td>
<td>15 .0165</td>
<td>3.5 .0026</td>
</tr>
<tr>
<td>LJ-150.283.4</td>
<td>10.0 .0496</td>
<td>0.0 .0024</td>
<td>27.5 .0736</td>
<td>1.0 .0160</td>
<td>1.5 .0027</td>
</tr>
<tr>
<td>LJ-150.283.10</td>
<td>13.5 .0525</td>
<td>0.0 .0026</td>
<td>29.5 .0774</td>
<td>2.5 .0206</td>
<td>1.0 .0029</td>
</tr>
<tr>
<td>LJ-150.283.15</td>
<td>15.0 .0530</td>
<td>0.0 .0027</td>
<td>29.5 .0781</td>
<td>2.0 .0174</td>
<td>0.5 .0029</td>
</tr>
<tr>
<td>LJ-40.283.4</td>
<td>19.5 .0530</td>
<td>2.5 .0146</td>
<td>6.0 .0053</td>
<td>5.5 .0107</td>
<td>7.5 .0141</td>
</tr>
<tr>
<td>LJ-80.283.4</td>
<td>10.5 .0463</td>
<td>1.0 .0064</td>
<td>1.5 .0046</td>
<td>15 .0054</td>
<td>3.0 .0056</td>
</tr>
<tr>
<td>LJ-120.283.4</td>
<td>15.5 .0518</td>
<td>0.5 .0041</td>
<td>0.5 .0038</td>
<td>0.0 .0030</td>
<td>2.5 .0055</td>
</tr>
<tr>
<td>LJ-160.283.4</td>
<td>10.5 .0472</td>
<td>0.0 .0014</td>
<td>1.0 .0044</td>
<td>0.0 .0016</td>
<td>0.0 .0016</td>
</tr>
<tr>
<td>LJ-200.283.4</td>
<td>17.5 .0535</td>
<td>0.0 .0006</td>
<td>0.0 .0006</td>
<td>0.0 .0006</td>
<td>0.0 .0006</td>
</tr>
<tr>
<td>LJ-240.283.4</td>
<td>13.0 .0480</td>
<td>0.0 .0006</td>
<td>0.0 .0006</td>
<td>0.0 .0007</td>
<td>0.0 .0007</td>
</tr>
<tr>
<td>LJ-280.283.4</td>
<td>15.0 .0532</td>
<td>0.0 .0004</td>
<td>0.0 .0004</td>
<td>0.0 .0005</td>
<td>0.0 .0003</td>
</tr>
<tr>
<td>Average</td>
<td>19.1 .0544</td>
<td>1.0 .0035</td>
<td>9.2 .0226</td>
<td>1.9 .0081</td>
<td>1.6 .0032</td>
</tr>
</tbody>
</table>

**Figure 1.** Results from using citation-kNN on the Affinity data set. In each of these plots the x-axis corresponds to the actual label and the y-axis gives the predicted label.

**Figure 2.** Results from using variations of the scaling options for Citation-kNN with real-valued labels on LJ-150.283.10.
used in generating the data. For example, on data set L.J-160.166.1, diverse density (when obtaining 100% accuracy) assigned very high scale values to some irrelevant features and some relevant features had low scale factors. Hence, one cannot use the ending scale factors to make any conclusion about the level of relevancy of a feature. In fact, for the Musk 1 data, by picking a different starting point, we obtained comparable results (namely, 87% accuracy) as Maron and Lozano-Perez’s results yet a completely different set of features had high scale factors. The feature with the highest scale factor in Maron’s work had one of the lowest in our classifier yet both worked equally well.

Next we show some plots to partly explain why such good results have been obtained on the musk data sets. We created three data sets, each of which has 166 features. To mimic the choice of data (strongly musk or strongly non-musk) in the Musk1 and Musk2 data sets, in L.J-160.166.1-S all bags that had labels between 0.3 and 0.6 were removed. We generated bags until we kept 92 of them. In L.J-160.166.1 no constraints were placed on the labels. Finally, to understand the effect of increasing the number of irrelevant features, we generated a third set (L.J-80.166.1-S) which was like the first but only 80 features were relevant. This data was generated using the LJ formula but computing the diverse density we used Maron’s formula with all initial scale factors set to 0.1. The results are shown in Figure 4. Notice how using the “strong” data set (leftmost plot) with a high number of relevant features made the learning task easier. The results are much worse when the data is not restricted to be only “strongly negative” and “strongly positive” (middle plot). Finally, if the number of irrelevant features rise then even with a “strong” data set, the results get noticeably worse (right plot).

7. Concluding Remarks
In this paper we present extensions of nearest-neighbor and diverse density algorithms for the real-valued setting. Our initial studies have provided some important insights into these algorithms. The performance of both the nearest neighbor and diverse density algorithms are very sensitive to the number of relevant features. When most of the features are relevant the performance is quite good and degrades as a larger fraction of the features become irrelevant. There is also some dependence on the number of different scale factors but this has a smaller effect in performance than the number of relevant features. (Both of these phenomena can be seen for the nearest neighbor algorithms in the results shown in Table 3.) We believe that good performance has been obtained on the musk data sets partly because a significant fraction of the features are relevant.

An important problem that must be addressed for both sets of algorithms is the development of a technique to accurately estimate which features are relevant. As discussed earlier, prior work using the musk data sets have made assumptions about which features were relevant based on the final scale factors found by the diverse density algorithm. Using artificial data we have shown that these values have basically no correlation to the real values. We believe the reason for this phenomena is that scale factors are found that increase performance for a local optimum in the gradient ascent search. We have provided a heuristic to estimate which features are relevant, but much more research in this direction is needed. Perhaps the diverse density search heuristic could be adapted to accelerate the search and obtain final scale factors that correlate better with the true scale factors. Having artificial data sets in which the true scale factors are known will be very valuable in evaluating the effectiveness of different algorithms to estimate which features are relevant.

Another interesting finding is that for 2-NN, 8-NN, and citation-kNN when the true scale factors are known, we actually obtained better results by projecting out the irrelevant features (with no other stretching of the axes) than when stretching the axes based on the true scale factors. Further research is needed to fully understand this phenomena.

<table>
<thead>
<tr>
<th>data set</th>
<th>Diverse Density</th>
<th>citation-kNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>%err loss</td>
<td>%err loss</td>
</tr>
<tr>
<td>L.J-160.166.1-S</td>
<td>0.0 .0052</td>
<td>0.0 .0022</td>
</tr>
<tr>
<td>L.J-160.166.1</td>
<td>23.9 .0852</td>
<td>4.3 .0014</td>
</tr>
<tr>
<td>L.J-80.166.1-S</td>
<td>53.3 .1116</td>
<td>0.0 .0025</td>
</tr>
<tr>
<td>L.J-16.30.2-0.9</td>
<td>6.7 .0074</td>
<td>8.3 .0197</td>
</tr>
<tr>
<td>L.J-16.30.2</td>
<td>6.7 .0240</td>
<td>16.7 .0260</td>
</tr>
<tr>
<td>Affinity</td>
<td>26.8 .0421</td>
<td>14.4 .0124</td>
</tr>
</tbody>
</table>

Table 4. Comparison of results of diverse density (where all scale factors are initially 0.1) and citation-kNN with no scaling when using real-valued labels.
search. For the nearest neighbor algorithms there are many variants which should be systematically studied such as using a weighted variant. We also need to further study the optimal choice for the number of references and citers for the citation-kNN algorithm and for $k$ for k-NN. Artificial data sets in which the number of points per bag is comparable to the affinity data set has been generated and we are running experiments on these data sets. In addition, we plan on comparing the algorithms we have considered here with COMPASS and an approach based on the EM algorithm. Finally, we would like to consider other application areas for studying real-valued multiple-instance learning.

References


