Multiple Instance Learning via Covariant Aggregation

Artificial Intelligence and Robotics Technical Report #1139

Thomas J. Palmer, Matthew Bodenhamer, Andrew H. Fagg
Symbiotic Computing Laboratory, University of Oklahoma, USA
tompalmer@ou.edu, mbodenhamer@ou.edu, fagg@cs.ou.edu

Abstract

We present a multiple instance learning (MIL) algorithm that learns ellipsoidal decision boundaries with arbitrary covariance. In contrast to the fixed-length feature vectors of traditional classification problems, MIL operates on unordered bags of instances. Commonly, each instance is a feature vector, and a bag is considered positive if any one of its instances is positive. In training data, bags are labeled, rather than the instances themselves. Existing MIL approaches either do not acknowledge covariances that exist in the feature space or do not provide simple descriptions of the decision volumes. Our method learns simple volumes via incremental aggregation of volume-describing instances from the positive bags. Overall, we show effective and robust handling on low-dimensional, covariant learning problems, as well as competitive performance on standard MIL data sets.

Introduction

Machine learning, including classification, usually addresses inputs of fixed-length feature vectors. In contrast, multiple instance learning (MIL) addresses unordered sets or bags of instances, where each instance is commonly a feature vector (Dietterich, Lathrop, and Lozano-Pérez 1997). Also, binary MIL classification problems commonly consider a bag *positive* if it has at least one instance that is a positive example of some concept. In this formulation, MIL is an existential classification problem:

$$positive(B) = \exists_{x \in B}\, positive(x), \tag{1}$$

where B is a bag, and each x is an instance. MIL therefore goes beyond ordinary vector-based learning, in that the learning algorithm must identify relevant instances while also learning to classify them.

Common examples of MIL include that of molecule classification where each molecule has a set of potential foldings, and any one of those foldings might signify the key behavior of the molecule. Image classification is another example, where the task is to identify whether or not an image contains a particular object, given a certain bag of visual features. As a third example, in playing a game of soccer, passing a ball is likely to fail if there exists an opponent between the passer and the potential receiver; positions of other opponents might be less relevant.

Early MIL algorithms include axis-parallel rectangles (APR) of Dietterich, Lathrop, and Lozano-Pérez (1997) and diverse density (DD) of Maron and Lozano-Pérez (1997). APR greedily selects discriminating features, choosing minimum and maximum bounds for each feature value to ensure that at least one instance from each positive bag is included. DD defines a probabilistic metric emphasizing instances with many positive and few negative bags nearby. Using the DD metric, gradient ascent selects the best central feature vector and the scaling for each feature. Many other MIL algorithms and multiple-instance problem formulations (beyond just existential queries) have since been developed (see Foulds and Frank 2010). These include such concerns as bag distance metrics and MIL-oriented kernels and constraints for support vector machines. While many of these techniques are often quite effective, learned models are not always intuitively clear to humans. Further, some techniques, including DD and APR, label individual instances as positive or negative, while others focus exclusively on labeling bags as wholes.

In our work, we seek to determine instance labels, as well as to provide a simple description of the relevant volume in the feature space, such as those provided by APR and DD. However, neither APR nor DD acknowledge covariance that can occur across features. Covariance can arise, for example, in color models (such as yellow in RGB space, which contains equal parts red and green) or in spatial configurations (where interesting cases have some object along a particular vector). In MIL, simply realigning data to principal components is not ideal, as the representative instances might align very differently than the aggregate set of positive instances. While Zhao et al. (2013) address learning of covariant distance metrics within MIL, they focus neither on simple decision volumes nor instance classification. In this paper, we present a learning technique that directly addresses the learning of simple, covariant decision volumes without the need to tune algorithm parameters for individual learning problems.

In the remainder of this paper, we describe our MIL algorithm, which learns a covariant decision volume via iterative aggregation of volume-describing instances from positive bags. We then evaluate our method, on both synthetic and real-world data sets, against other well-known algorithms (including DD) that either describe ellipsoidal regions or else use radial feature kernels.

Procedure 1 Covariant aggregation summary

Begin set K with one initial key point.
Let B_R be all positive bags not containing initial point.
Estimate μ and V from K.
Let W be witness points of min $D_M(x|\mu, V)$ for all bags.
Choose radius r to maximize training set accuracy.
Remember (μ, V, r) as initial best.
While progress seems likely do:
  Let B_S be M sampled bags from B_R.
  For each B in B_S do:
    Let K' be $K \cup$ the witness point from W for B.
    Determine μ', V', W', and r' using K'.
    Evaluate training accuracy for K'.
  Update K, μ, V, W, and r for best B sampled.
  Update B_R to exclude best B.
  If new training accuracy $\geq$ previous best do:
    Remember new best (μ, V, r).
Return best (μ, V, r).

Learning Algorithm

We follow the common existential, instance-classifying MIL formulation established in Equation 1. In this form, the classifier's job is to classify individual instances. If any instance in a bag is found to be positive, then the bag itself is labeled as positive.

The differentiating aspect of our approach is that of covariant decision volumes. This contrasts with the axis-aligned volumes of APR and DD. Specifically, we describe a decision volume by its mean μ, covariance V, and radius r. Parameters μ and V provide the Mahalanobis distance from the volume center:

$$D_M(x|\mu, V) = \sqrt{(x-\mu)^T V^{-1}(x-\mu)}. \tag{2}$$

Any instance x within radius r is considered positive:

$$positive(x|\mu, V, r) \iff D_M(x|\mu, V) \leq r. \tag{3}$$

Throughout our discussion, we treat instances as feature vectors within a Euclidean topology. As such, we also refer to instances as *points*.

We now describe an algorithm for learning covariant decision volumes from training data with labeled bags. Procedure 1 gives informal pseudocode for the learning process, and Figure 1 shows several steps of the process using a synthetic data set in which positive bags are more likely to have at least one instance along the diagonal. In summary, our algorithm seeks a set of *key points*, K, from which to estimate parameters μ and V. This process begins with a single key point from one sampled positive bag. At each iteration, the algorithm chooses a new key point from an unrepresented positive bag to add to this set. Given the new potential set of key points, μ and V are reestimated, and a new radius, r, is chosen to maximize the training accuracy (similar to the method of Auer and Ortner 2004). Key point aggregation continues while training set accuracy across recent iterations suggests possible improvement.

In the remainder of this section, we elaborate on each step of the algorithm.

Covariance Estimation

Our algorithm determines a covariant, ellipsoidal decision volume starting from a single key point. Additional key points are aggregated iteratively. We calculate μ as the mean of the key points. However, when there are few key points relative to the dimensionality, covariance V is poorly defined. To condition the covariance, we begin with an inverse Wishart prior based on all points from all positive bags. The inverse Wishart distribution is conjugate prior to the multivariate covariance matrix (see Gelman et al. 2013). That is, given additional observations defining a covariance matrix, the posterior distribution is still inverse Wishart.

The inverse Wishart distribution has two parameters: a scale matrix, Ψ, and degrees of freedom, ν. The matrix Ψ represents a prior observation of covariance scaled by ν presumed observations. The posterior Ψ_{post} is given by:

$$\Psi_{post} = \mathrm{ncov}(X) + \Psi$$

where

$$\mathrm{ncov}(X) = (X - E[X])(X - E[X])^T \tag{4}$$

is a scale matrix based on observations as column vectors in the matrix X.

Usually, the magnitude of the posterior covariance depends on the sum of ν and the number of new observations. However, in our case, we use our estimate of V only for the shape and orientation of the decision volume. Radius r is determined at a later step.

Also, because our learning problem is multiple-instance, there is no necessary relationship between all instances from positive bags and the final decision volume. Therefore, we treat the weight of the prior as independent of both the number of bags and the number of instances. Instead, we use a hand-selected parameter, w, to influence the overall weight of the prior. Our effective posterior covariance is therefore:

$$V_{post} = \mathrm{ncov}(K) + w\,\mathrm{ncov}(P)/p, \tag{5}$$

where K represents the key points and P represents all points (quantity p) from all positive bags. Figures 1(a) and 1(b) show an initial covariance based entirely on the prior, while Figures 1(c) and 1(d) show the effect of an increasing number of key points.

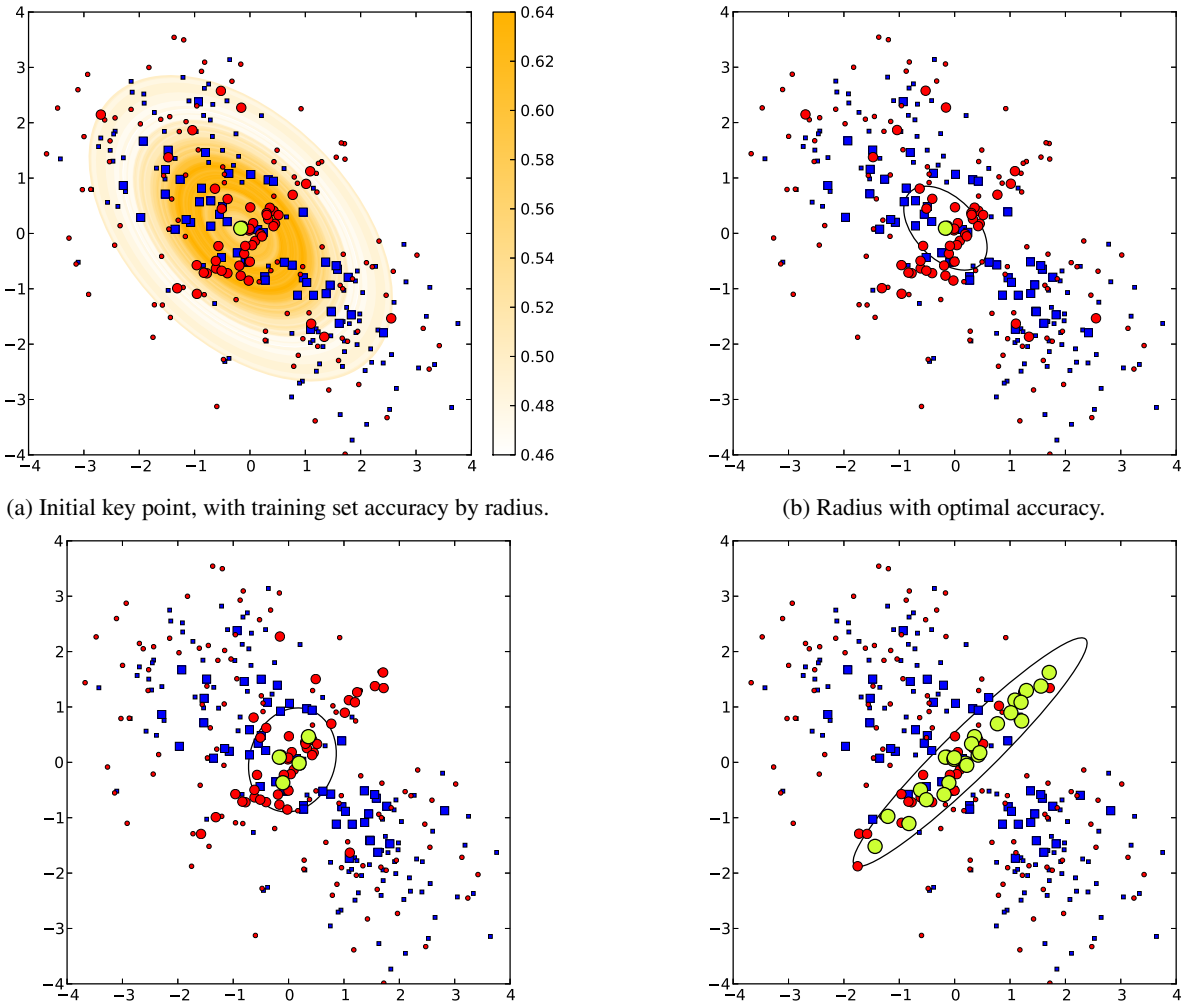
However, the covariance could still be poorly conditioned, especially in situations of high dimensionality. If the condition number of the covariance matrix is above a particular threshold c, we add a constant diagonal to this covariance:

$$V = \begin{cases} V_{post} & \text{if } \left|\frac{\max \Lambda}{\min \Lambda}\right| \leq c \\ V_{post} + \frac{\max \Lambda - c \min \Lambda}{c-1} I & \text{otherwise,} \end{cases} \tag{6}$$

where Λ is the set of eigenvalues of V_{post}, and I is the identity matrix. This yields a covariance matrix with a maximum condition number of c.

Decision Volume Radius

To choose decision volume radius r, we follow the optimal ball algorithm of Auer and Ortner (2004). Specifically, we select the radius that maximizes classification accuracy



(a) Initial key point, with training set accuracy by radius.

(b) Radius with optimal accuracy.

(c) Decision volume after four key points.

(d) At 24 key points, the final decision volume.

Figure 1: Overview of the learning process. Circles indicate points from positive bags, and squares indicate negatives. Pale circles indicate key points. Larger markers indicate witness points: the nearest point from each bag, given the current mean and covariance.

across all bags in the training set. Because each bag is classified as to whether any of its points lie within the decision volume, only the nearest point to the volume center is of interest. This nearest point is called the *witness point* for its bag. We differ from Auer and Ortner primarily in our use of Mahalanobis distance. Formally, our set of witness points W is defined as:

$$W = \left\{ \operatorname*{argmin}_{x \in B} D_M(x|\mu, V) \,\middle|\, B \in \mathcal{B} \right\}, \qquad (7)$$

where $\mathcal{B}$ represents all training bags (both positive and negative).

To choose radius r, we first sort W by increasing distance, and then perform a linear scan for maximal accuracy, also as suggested by Auer and Ortner. Figure 1(a) illustrates training set accuracy as a function of radius, and Figure 1(b) shows the corresponding radius maximizing accuracy.

Key Point Aggregation

To begin the learning process, our algorithm samples a single positive bag. For each point in the bag, using only the covariance prior as V, a radius is chosen to maximize training accuracy. Iterative covariant aggregation, shown in Figure 1, begins from the point yielding highest accuracy.

At each aggregation step, the algorithm samples M positive bags currently unrepresented among the key points, where M is an algorithm parameter. Further, to keep the search localized, only bags whose witness points lie inside the current decision volume are sampled. The witness point (and no others) from a sampled bag is tentatively added to the key points, and a new decision volume is constructed. Among sampled bags, the tentative key point set with highest accuracy is selected, and the aggregation process repeats.

Early in the search process, with few key points, the decision volume's shape is very flexible. It becomes more es-

tablished as the number of key points increases. Therefore, to terminate the aggregation process, we make a linear least squares estimate of training accuracy over the most recent A aggregation steps, where A is an algorithm parameter. We continue aggregation if the accuracy slope is positive and if, continuing the current slope through remaining positive bags, the accuracy would surpass the previous best (including across restarts, as discussed shortly). Otherwise, aggregation stops, retaining the model with the highest accuracy. If decision volumes at multiple iterations have equal accuracy, we select the one with the most key points, as this presumably better describes the volume. At minimum, a full window of A iterations is required before termination.

As is common for stochastic algorithms, we restart the learning process multiple times. After each aggregation process, we sample another positive bag for the next restart. Algorithm parameter N determines the number of restarts. Of the N resulting models, the volume with the highest accuracy with respect to the training set (or, in case of a tie, the one with the most key points) is selected as the final volume.

Experimental Evaluation

We test our method on two types of data sets: (1) synthetic data in two or three dimensions, of which some exhibit covariant regions of interest (discussed below), and (2) standard third-party MIL data sets, which usually have high dimensionality and do not necessarily exhibit interesting covariance.

We compare against other algorithms with ellipsoidal decision boundaries or ellipsoidal kernels. Techniques with ellipsoidal boundaries include the non-boosted optimal ball algorithm of Auer and Ortner (2004), and the DD (Maron and Lozano-Pérez 1997) and EM-DD (Zhang and Goldman 2001) algorithms. None of these are capable of expressing covariance, although DD and EM-DD scale original axes. We use the Weka (Hall et al. 2009) implementations of these algorithms with default parameters.

We also compare against SVM techniques mi-SVM, MI-SVM (Andrews, Tsochantaridis, and Hofmann 2002), and MILES (Chen, Bi, and Wang 2006). For mi-SVM and MI-SVM, we use (isotropic) RBF kernels, and the MILES feature set is also based on a radially decaying measure. All three are therefore capable of expressing arbitrarily nonlinear surfaces, including approximations of covariance. Unlike other techniques compared here, MILES is a bag-only classifier rather than an instance classifier, but we include it because it is simple and performs well. We use the MISVM package of Doran and Ray (2013) for implementations of mi-SVM and MI-SVM. We use our own implementation of the MILES feature set with LIBLINEAR (Fan et al. 2008) as the SVM classifier, using the L2-loss L1-regularization option, which differs from the L1-loss of standard MILES.

In testing the algorithms, we are primarily concerned with performance "in the wild." That is, we assume that new learning problems need to be addressed without the opportunity for extensive analysis and parameter tuning. We therefore hold all parameters constant for our learning algorithm. Specifically, we hold constant the progress window size A at 8, covariance prior weight w at 10^{-1}, max covariance condition c at 10^5, and both number of restarts N and number of aggregations bags sampled M at 4. We have chosen these parameters based upon exploratory investigations with some of the synthetic data sets, as well as the well-known Musk 1 data set (Dietterich, Lathrop, and Lozano-Pérez 1997).

Using a similar amount of exploration, we have also selected parameters C and γ for evaluation of SVM-based techniques. For C, we use a constant value of 10^4. We choose $\gamma = 1/2\sigma^2$ using a heuristic calculation for σ as the mean of the tenth and ninetieth percentiles of all (positive and negative) inter-point distances in the training set (e.g., as used as a starting point by Takeuchi et al. 2006).

Also, in exploration with DD and EM-DD, we found it necessary to filter the Musk data sets for meaningful learning results. Specifically, we divide the values of each feature dimension by their standard deviation. For consistency, we do this for all data sets for DD and EM-DD. We have not performed such filtering for other learning methods.

For our synthetic data sets, in addition to the heuristically-chosen SVM parameters, we also test against tuned SVM parameters. In these cases, we search across a grid of C values from 10^{-4} to 10^9 (stepping by powers of 10) and factors of our heuristically-chosen γ, from 2^{-7} to 2^7.

Our reported results use different seeds for data generation, shuffling, and/or stochastic learning than those used during exploratory evaluation or parameter selection.

Synthetic, Low-Dimensional Data Sets

Our method is designed for use in low-dimensional (often physical) settings with potentially covariant classification volumes. We have therefore designed multiple synthetic learning problems that exhibit interesting characteristics in two or three dimensions. These data sets consist of the following mixture distributions:

- **Covariant 1**, depicted in Figure 1, is a 2D problem where negative instances come from one of two isotropic Gaussians of standard deviation 1 and means of $(-2, 2)$ and $(2, -2)$, respectively. Positive instances come from a covariant Gaussian along the orthogonal diagonal, centered at $(0, 0)$. Eigenvalues of the positive distribution are 1 and $1/5^2$. All bags have 3 instances, where exactly 1 of the 3 is drawn from the positive distribution for each positive bag.
- **Covariant 2** is the same as Covariant 1, except that there is a single negative distribution, centered at $(0, 0)$ along with the positive distribution. This problem is therefore very noisy.
- **Color** is a 3D problem where each distribution represents a red, green, blue, or yellow color in RGB space. The distributions are based on photographs of printed color patterns, and thus represent data with real-world characteristics. Red, green, and blue are negative distributions. Yellow is positive and is inherently covariant in RGB space. All bags contain 5 instances. Positive bags again contain exactly one instance drawn from the positive distribution.
- **Disjunctive** is a non-covariant data set, designed to challenge our proposed approach by drawing positive in-

stances from a non-compact set. A single negative, isotropic distribution is centered at $(0, 0)$ with a standard deviation of 1. The positive distribution is a mixture of two, isotropic Gaussian distributions, centered at $(\pm 3, 0)$, each with a standard deviation $\sqrt{1/2}$. All bags contain ten instances, filling up the negative space thoroughly. Positive bags contain exactly one instance drawn from either side of the mixture (both are equally probable).

For all synthetic data sets in this work, we generate 100 training bags (50 positive and 50 negative) and 100 test bags. Because the data is synthetic, rather than doing n-fold cross-validation, we generate 100 independent training and test sets for calculating statistics.[1]

Standard MIL Data Sets

In addition to our synthetic cases, we also test using several standard data sets. Specifically, we use MIL data sets provided by the Weka project, including most of those evaluated by Foulds and Frank (2008). We specifically evaluate performance on the well-known data sets Musk 1 and Musk 2; other chemical classification sets Thioredoxin and Mutagenesis Atoms, Bonds, and Chains; the Corel image data sets Elephant, Fox, and Tiger; and the train direction classification problem East-West. From the Weka-distributed sample files, we exclude the Component, Function, and Process data sets because they are very large. We also exclude two data sets for applicability reasons: Suramin, because some instances contain unspecified values and West-East, because it is a universal rather than an existential problem.

Of the third-party data sets we test, the least number of dimensions (for Thioredoxin) is 8. Some data sets (such as Fox or Musk) have more than one hundred. Furthermore, we have no expectation in advance of which data sets are well represented by covariant volumes. As such, these standard sets are outside the designed use case of our method. Still, we include test results here for comparison with other MIL algorithms. For synthetic data sets, statistics are computed using stratified 10-fold cross validation.

Evaluation Method

We perform statistical comparisons using a standard two-sample, two-tailed t-test using $\alpha = 0.05$. For cases where the mean score of our method is better than that of another, we additionally apply Šidák correction for multiple comparisons:[2] $\alpha_{corrected} = 1-(1-\alpha)^{1/n}$. For our 9 comparisons, $\alpha_{corrected} \approx 0.00568$. When another method outperforms ours, we retain the original α. This has the effect of not overestimating the performance of our approach, whether our approach performs better or worse than another.

We report algorithm performance using the Peirce Skill Score (PSS), which penalizes merely reporting the most common label (Stephenson 2000). In binary classification, this score is equivalent to the hit rate minus the false alarm rate. A PSS of 1 is perfect, 0 is random or majority class assignment, and -1 is a perfectly incorrect labeling. Many of our evaluated data sets have balanced or nearly-balanced labels. Therefore, higher PSS here often implies higher accuracy, but not in every case.

[1] For blind review, we include generated data as a submission attachment. For the final publication, we plan to provide a URL.

[2] Jensen and Cohen (2000) refer to this correction as Bonferroni adjustment.

Results

As shown in Figure 2, our method, labeled CovAgg, outperforms the other methods, with the exception of MI-SVM, on the three explicitly covariant data sets. Across these sets, MI-SVM performs about as well as our approach. As further shown in Figure 4, the default parameter values for MI-SVM on these sets are already near the best performing parameters. Parameter tuning brings mi-SVM to nearly the same performance level. It therefore also seems that, even in noisy settings, our method extracts covariant volumes very effectively. As expected, Optimal ball ("OptBall" in the table), DD, and EM-DD fail to support covariance, as exhibited by the low performance.

On the other hand, all methods seem to do fairly well on the Disjunctive set, perhaps with the exception of EM-DD. While CovAgg performs below the level of some others, the difference isn't great. Most of the techniques, including CovAgg, extract volumes covering one of the two positive areas. Notably, mi-SVM and MILES-LL seem capable of extracting both areas. It is unsurprising that an SVM with an RBF kernel could accomplish this. Interestingly, MI-SVM, even with parameter tuning, is unable to find both positive regions.

Across the standard, third-party data sets, CovAgg performs approximately as well as the other methods. It does fall behind most techniques on Tiger and behind MI-SVM and MILES-LL on Elephant. The SVM techniques, on the other hand, perform poorly overall on the Mutagenesis data sets using the default parameters. That is, our heuristic SVM parameter selection works well for some problems, but tuning is needed for others.

Overall, the SVM techniques compare in performance to ours on covariant problems, but only if tuned properly. Across the board, the simpler techniques (such as CovAgg and Optimal Ball) are less likely to completely degenerate, as do the SVM methods on the Mutagenesis data sets.

Further, Figure 3 shows CPU user-space execution time. At least for these implementations, mi-SVM and MI-SVM are usually fast but are occasionally very slow at converging. In the case of Thioredoxin, the mi-SVM and MI-SVM runs failed to finish (cut off at over 24 CPU hours). Tuning SVM parameters requires additional time, unless this cost is amortized in prior analysis of a data set. Finally, we also find that DD, as reported by Zhang and Goldman (2001), can be very slow at times. It should also be noted that each training and testing of Weka-based implementations includes overhead Java VM startup time, which is probably most notable in the faster-running synthetic data sets.

Discussion

We have presented a method for multiple instance learning of covariant volumes that provides labels for indiviu-

	CovAgg	OptBall	DD	EM-DD	mi-SVM	MI-SVM	MILES-LL
Color	0.93 ± 0.01	$> 0.88 \pm 0.01$	$> 0.88 \pm 0.01$	$> 0.88 \pm 0.03$	$> 0.78 \pm 0.03$	$< 0.95 \pm 0.01$	$> 0.86 \pm 0.01$
Covariant 1	0.76 ± 0.02	$> 0.46 \pm 0.02$	$> 0.44 \pm 0.02$	$> 0.38 \pm 0.04$	$> 0.39 \pm 0.03$	0.73 ± 0.03	$> 0.24 \pm 0.03$
Covariant 2	0.27 ± 0.02	$> 0.13 \pm 0.02$	$> 0.06 \pm 0.02$	$> 0.10 \pm 0.02$	$> 0.11 \pm 0.02$	$> 0.21 \pm 0.02$	$> 0.07 \pm 0.01$
Disjunctive	0.27 ± 0.02	$< 0.33 \pm 0.02$	$< 0.31 \pm 0.02$	$> 0.19 \pm 0.04$	$< 0.62 \pm 0.02$	0.24 ± 0.06	$< 0.47 \pm 0.03$
Musk 1	0.61 ± 0.18	0.51 ± 0.22	0.71 ± 0.17	0.71 ± 0.19	0.64 ± 0.15	0.60 ± 0.21	0.63 ± 0.12
Musk 2	0.56 ± 0.11	0.49 ± 0.14	0.58 ± 0.23	0.69 ± 0.15	0.43 ± 0.18	0.49 ± 0.16	0.59 ± 0.19
Muta Atoms	0.54 ± 0.14	0.47 ± 0.19	0.34 ± 0.21	0.20 ± 0.20	> 0.00	$> 0.17 \pm 0.15$	$> 0.09 \pm 0.15$
Muta Bonds	0.46 ± 0.15	0.45 ± 0.20	0.36 ± 0.15	0.35 ± 0.15	> 0.00	> 0.00	$> 0.15 \pm 0.11$
Muta Chains	0.42 ± 0.14	0.31 ± 0.14	0.50 ± 0.18	0.23 ± 0.23	> 0.00	0.28 ± 0.15	0.47 ± 0.17
Thioredoxin	0.13 ± 0.14	0.30 ± 0.21	0.34 ± 0.28	0.18 ± 0.22	-	-	0.00
East West	0.00 ± 0.45	$< 0.60 \pm 0.35$	0.10 ± 0.50	0.20 ± 0.43	0.00	0.40 ± 0.47	0.30 ± 0.33
Elephant	0.46 ± 0.10	0.58 ± 0.12	$< 0.60 \pm 0.09$	0.50 ± 0.07	0.58 ± 0.13	$< 0.71 \pm 0.09$	$< 0.66 \pm 0.11$
Fox	0.15 ± 0.09	$> -0.06 \pm 0.08$	0.29 ± 0.13	0.25 ± 0.17	0.12 ± 0.11	0.10 ± 0.13	0.18 ± 0.11
Tiger	0.17 ± 0.14	0.23 ± 0.13	$< 0.47 \pm 0.15$	$< 0.44 \pm 0.14$	$< 0.58 \pm 0.11$	$< 0.66 \pm 0.10$	$< 0.52 \pm 0.11$

Figure 2: Mean PSS, using constant/heuristic parameter choices for each learning method. Bounds, where variance occurred, represent 95% confidence intervals presuming a t-distribution. Inequalities, where present, indicate a statistical difference between CovAgg and the indicated method. The symbol '-' indicates aborted runs.

	CovAgg	OptBall	DD	EM-DD	mi-SVM	MI-SVM	MILES-LL
Color	0.47 ± 0.01	$< 1.18 \pm 0.02$	$< 4.23 \pm 0.16$	$< 1.34 \pm 0.02$	$< 6.04 \pm 1.14$	0.49 ± 0.02	$> 0.19 \pm 0.00$
Covariant 1	0.57 ± 0.02	$< 0.92 \pm 0.01$	$< 1.77 \pm 0.06$	$< 0.85 \pm 0.01$	$< 2.96 \pm 0.46$	$> 0.20 \pm 0.01$	$> 0.15 \pm 0.00$
Covariant 2	0.44 ± 0.01	$< 1.01 \pm 0.01$	$< 2.20 \pm 0.09$	$< 0.93 \pm 0.02$	$< 1.54 \pm 0.33$	$> 0.28 \pm 0.03$	$> 0.16 \pm 0.00$
Disjunctive	0.44 ± 0.01	$< 1.31 \pm 0.01$	$< 10.9 \pm 0.5$	$< 1.47 \pm 0.01$	$< 32.6 \pm 5.2$	$< 6.22 \pm 0.70$	$< 0.58 \pm 0.00$
Musk 1	10.2 ± 0.3	$> 2.25 \pm 0.07$	$< 16.6 \pm 4.0$	$< 22.5 \pm 5.4$	$> 0.56 \pm 0.15$	$> 0.43 \pm 0.08$	$> 0.32 \pm 0.02$
Musk 2	36.0 ± 5.3	$> 10.6 \pm 0.4$	$< 1358 \pm 527$	$< 265 \pm 111$	$< 4157 \pm 1244$	$< 1557 \pm 199$	$< 47.1 \pm 3.6$
Muta Atoms	0.75 ± 0.04	$< 2.29 \pm 0.05$	$< 7.31 \pm 0.73$	$< 1.96 \pm 0.07$	$< 9.74 \pm 1.77$	$< 5.52 \pm 0.72$	$< 1.33 \pm 0.05$
Muta Bonds	1.26 ± 0.08	$< 4.25 \pm 0.10$	$< 32.1 \pm 18.1$	$< 3.68 \pm 0.46$	$< 309 \pm 56$	$< 15.1 \pm 1.3$	$< 5.14 \pm 0.06$
Muta Chains	2.08 ± 0.14	$< 6.08 \pm 0.12$	$< 125 \pm 25$	$< 5.43 \pm 0.68$	$< 1775 \pm 321$	$< 27.4 \pm 5.4$	$< 9.30 \pm 0.17$
Thioredoxin	4.21 ± 0.49	$< 8.56 \pm 0.45$	$< 681 \pm 68$	$< 14.4 \pm 1.7$	-	-	$< 121 \pm 2$
East West	0.24 ± 0.01	$< 0.80 \pm 0.03$	$< 2.91 \pm 0.31$	$< 1.42 \pm 0.05$	$> 0.09 \pm 0.03$	$> 0.05 \pm 0.01$	$> 0.02 \pm 0.00$
Elephant	24.3 ± 2.9	$> 4.58 \pm 0.13$	$< 219 \pm 32$	$< 155 \pm 22$	$< 70.0 \pm 28.4$	$> 5.46 \pm 0.92$	$> 5.04 \pm 0.10$
Fox	16.8 ± 1.5	$> 4.27 \pm 0.04$	$< 126 \pm 17$	$< 92.0 \pm 34.3$	$< 86.8 \pm 26.8$	$> 7.44 \pm 1.95$	$> 3.50 \pm 0.02$
Tiger	19.3 ± 2.3	$> 4.59 \pm 0.16$	$< 72.9 \pm 14.5$	$< 70.4 \pm 20.2$	53.5 ± 26.5	$> 5.71 \pm 0.64$	$> 3.47 \pm 0.06$

Figure 3: Mean training and prediction time in seconds, using constant/heuristic parameter choices for each learning method. Bounds, where variance occurred, represent 95% confidence intervals presuming a t-distribution. Inequalities, where present, indicate a statistical difference between CovAgg and the indicated method. The symbol '-' indicates aborted runs.

	mi-SVM*	MI-SVM*	MILES-LL*
Color	0.92 ± 0.01	$< 0.95 \pm 0.01$	$> 0.90 \pm 0.01$
Covariant 1	0.76 ± 0.02	0.78 ± 0.01	0.73 ± 0.02
Covariant 2	$> 0.20 \pm 0.02$	0.22 ± 0.03	$> 0.22 \pm 0.02$
Disjunctive	$< 0.65 \pm 0.02$	$< 0.35 \pm 0.06$	$< 0.56 \pm 0.02$

Figure 4: Mean PSS, using SVM parameters chosen by grid search. Bounds represent 95% confidence intervals presuming a t-distribution. Inequalities, where present, indicate a statistical difference relative to CovAgg.

dal instances in a bag and uses a simple, parametric representation for the decision volume. Without performing problem-specific parameter tuning, our approach performs particularly well relative to other, standard MIL approaches on problems in which the data exibit covariant properties. Our approach also performs competitively on typical MIL data sets. As such, we believe our algorithm to be a viable and effective MIL approach, especially when robustness to covariance is required, and when parameter turning is not a practical option.

Going forward, we are interested in the application of MIL methods to learning in multi-attribute, relational settings (e.g., Blockeel and De Raedt 1998; Bodenhamer et al. 2009). In these contexts, the MIL classifier is applied repeatedly for each potential relational question, and must be efficient and robust in finding effective decision volumes. While algorithms such as MI-SVM could perform better for some specific problems, the additional time required to select problem-specific parameters and the potential complexity of the resulting decision volumes could yield these alternative approaches ineffective in the broader relational settings.

In future work, we expect to apply our method to non-Euclidean topologies, such as orientation in two or three dimensions. While we have emphasized *covariant* aggregation here, the technique is easily applicable to other distance metrics.

References

Andrews, S.; Tsochantaridis, I.; and Hofmann, T. 2002. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems (NIPS)*, 577–584.

Auer, P., and Ortner, R. 2004. A boosting approach to multiple instance learning. In *European Conference on Machine Learning (ECML)*, volume 3201 of *Lecture Notes in Computer Science (LNCS)*. Springer Berlin / Heidelberg. 63–74.

Blockeel, H., and De Raedt, L. 1998. Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101(1–2):285–297.

Bodenhamer, M.; Bleckley, S.; Fennelly, D.; Fagg, A. H.; and McGovern, A. 2009. Spatio-temporal multi-dimensional relational framework trees. In *IEEE International Conference on Data Mining (ICDM) Workshop on Spatial and Spatiotemporal Data Mining (SSTDM)*, 564–570.

Chen, Y.; Bi, J.; and Wang, J. Z. 2006. MILES: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28(12):1931–1947.

Dietterich, T. G.; Lathrop, R. H.; and Lozano-Pérez, T. 1997. Solving the multiple instance learning problem with axis-parallel rectangles. *Artificial Intelligence* 89(1–2):31–71.

Doran, G., and Ray, S. 2013. A theoretical and empirical analysis of support vector machine methods for multiple-instance classification. *Machine Learning* (in press).

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research (JMLR)* 9:1871–1874.

Foulds, J., and Frank, E. 2008. Revisiting multiple-instance learning via embedded instance selection. In *Australasian Joint Conference on Artificial Intelligence*, volume 5360 of *Lecture Notes in Computer Science (LNCS)*. Springer Berlin Heidelberg. 300–310.

Foulds, J., and Frank, E. 2010. A review of multi-instance learning assumptions. *The Knowledge Engineering Review* 25(1):1–25.

Gelman, A.; Carlin, J. B.; Stern, H. S.; Dunson, D. B.; Vehtari, A.; and Rubin, D. B. 2013. *Bayesian Data Analysis*. Chapman and Hall/CRC.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The WEKA data mining software: An update. *SIGKDD Explorations* 11(1):10–18.

Jensen, D. D., and Cohen, P. R. 2000. Multiple comparisons in induction algorithms. *Machine Learning* 38(3):309–338.

Maron, O., and Lozano-Pérez, T. 1997. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems (NIPS)*, 570–576.

Stephenson, D. B. 2000. Use of the "odds ratio" for diagnosing forecast skill. *Weather and Forecasting* 15(2):221–232.

Takeuchi, I.; Le, Q. V.; Sears, T. D.; and Smola, A. J. 2006. Nonparametric quantile estimation. *Journal of Machine Learning Research (JMLR)* 7:1231–1264.

Zhang, Q., and Goldman, S. A. 2001. EM-DD: An improved multiple-instance learning technique. In *Advances in Neural Information Processing Systems (NIPS)*, volume 2, 1073–1080.

Zhao, H.; Cheng, J.; Jiang, J.; and Tao, D. 2013. Multiple instance learning via distance metric optimization. In *IEEE International Conference on Image Processing (ICIP)*, 2617–2621.