Bayesian Multiple Instance Learning: Automatic Feature Selection and Inductive Transfer

Vikas C. Raykar VIKAS.RAYKAR@SIEMENS.COM
Balaji Krishnapuram BALAJI.KRISHNAPURAM@SIEMENS.COM
Jinbo Bi JINBO.BI@SIEMENS.COM
Murat Dundar MURAT.DUNDAR@SIEMENS.COM
R. Bharat Rao BHARAT.RAO@SIEMENS.COM
CAD and Knowledge Solutions (IKM CKS), Siemens Medical Solutions Inc., Malvern, PA 19355 USA

Abstract

We propose a novel Bayesian multiple instance learning (MIL) algorithm. This algorithm automatically identifies the relevant feature subset, and utilizes inductive transfer when learning multiple (conceptually related) classifiers. Experimental results indicate that the proposed MIL method is more accurate than previous MIL algorithms and selects a much smaller set of useful features. Inductive transfer further improves the accuracy of the classifier as compared to learning each task individually.

1. Multiple Instance Learning

In a single instance learning scenario we are given a training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ containing N instances, where $x_i \in \mathcal{X}$ is an instance (the feature vector) and $y_i \in \mathcal{Y} = \{0, 1\}$ is the corresponding known label. The task is to learn a classification function $f : \mathcal{X} \rightarrow \mathcal{Y}$.

In the *multiple instance learning* framework the training set consists of *bags*. A bag contains many instances. All the instances in a bag share the same bag-level label. A bag is labeled positive if it contains *at least* one positive instance. A negative bag means that *all* instances in the bag are negative. The goal is to learn a classification function that can predict the labels of unseen instances and/or bags.

MIL is a natural framework to model many real-life tasks like drug activity prediction (Dietterich et al., 1997), image retrieval (Andrews et al., 2002), face detection (Viola et al., 2006), scene classification, text categorization, etc and is often found to be superior than a conventional supervised learning approaches (Ray & Craven, 2005). The concept of MIL was first introduced by (Dietterich et al., 1997) in the context of drug activity prediction. (Maron & Lozano-Perez, 1998) proposed a framework called Diverse Density algorithm. Since then various variants of standard single instance learning algorithms like Boosting (Xin & Frank, 2004; Viola et al., 2006), SVMs (Andrews et al., 2002; Fung et al., 2007), Logistic Regression (Ray & Craven, 2005; Settles et al., 2008), nearest neighbor (Wang & Zucker, 2000) etc. have been modified to adapt to the MIL scenario.

Our motivation for this work comes from the area of computer aided diagnosis (CAD) (see section § 10)– where the task is to build a classifier to predict whether a suspicious region (instance) on a computed tomography (CT) scan is a pulmonary embolism/nodule/lesion or not. This was proposed as a MIL problem by (Fung et al., 2007) by recognizing the fact that all instances which are within a certain distance to a radiologist mark (ground truth) can be considered as a positive bag. A requirement is that run time of the classifier during testing be as small as possible. Hence we would like the final classifier to use *as few features as possible*.

In this paper we propose a novel multiple instance algorithm which performs *automatic feature selection* and *classifier design* jointly. In particular we start out with the well-known logistic regression as our classifier and demonstrate how it can be modified for the MIL framework (§ 3–6). We use the feature selection method originally proposed for the *relevance vector machine* (RVM) (Tipping, 2001) single-instance classifier, in a manner that is optimal for multiple-instance classification(§ 7). We extend the algorithm to handle multi-task learning in § 8.

Appearing in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008. Copyright 2008 by the author(s)/owner(s).

2. Novel Contributions

Our method differs from the substantial body of previous literature on MIL in the following crucial aspects.

a. **Baseline-model:** We use Logistic Regression as our baseline (single instance) classifier, similar to two previous papers. However, our model for combining the positive instances is quite different from the soft-max (Ray & Craven, 2005) or the averaging approach (Xin & Frank, 2004) used by others. We directly enforce the definition that at least one of the instances in a positive bag is positive.

b. **Feature selection:** Relying on the Bayesian automatic relevance determination paradigm, our learning algorithm selects the relevant subset of features that is most useful for accurate *multiple instance* classification. Experimental results demonstrate that the number of features chosen for optimizing the accuracy of multiple-instance classification *is much smaller* than that selected in a corresponding single instance learning algorithm. While MI Boost (Xin & Frank, 2004) also does feature selection, results indicate that our our approach is more accurate than MI Boost.

c. **Inductive transfer:** The proposed method is easily extended to statistically exploit information from other data sets while learning multiple related classifiers. This inductive-transfer approach results in substantial improvements in accuracy in real-life problems with limited training data. We are not aware of previous work which accomplishes inductive transfer in the context of multiple-instance classification.

3. Notation

We represent an instance as a feature vector $x \in \mathbb{R}^d$. A bag which contains K instances is denoted by boldface $\mathbf{x} = \{x_j \in \mathbb{R}^d\}_{j=1}^K$. The label of a bag is denoted by $y \in \{0, 1\}$.

Training Data The training data $\mathcal{D}$ consists of N bags $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i = \{x_{ij} \in \mathbb{R}^d\}_{j=1}^{K_i}$ is a bag containing K_i instances that share the same label $y_i \in \{0, 1\}$.

Classifier We consider the family of linear discriminating functions: $\mathcal{F} = \{f_w\}$, where for any $x, w \in \mathbb{R}^d$, $f_w(x) = w^T x$. The final classifier can be written in the following form

$$y = \begin{cases} 1 & \text{if } w^T x > \theta \\ 0 & \text{if } w^T x < \theta \end{cases} . \tag{1}$$

Ties are resolved by flipping a fair coin. The *threshold parameter* θ determines the operating point of the classifier. The ROC curve is obtained as θ is swept from $-\infty$ to ∞. A bag is labeled positive if at least one instance is positive and negative if all instances are negative. Learning a classifier implies choosing the weight vector w given the training data $\mathcal{D}$.

4. Logistic Generalized Linear Model

The posterior probability for the positive class is modeled as a *logistic sigmoid* acting on the linear classifier f_w, *i.e.*, $p(y = 1|x) = \sigma(w^\top x)$. The logistic sigmoid function is defined as $\sigma(z) = 1/(1 + e^{-z})$. This classification model is known as *logistic regression* in the statistics community. Also $p(y = 0|x) = 1 - p(y = 1|x) = 1 - \sigma(w^\top x)$.

4.1. Logistic Model for MIL

In the MIL framework we have the concept of bags– where all the examples in a bag share the same label. A positive bag means at least one example in the bag is positive. The probability that a bag contains at least one positive instance is one minus the probability that all of them are negative. Hence the posterior probability for the positive bag can be written as

$$p(y = 1|\mathbf{x}) = 1 - \prod_{j=1}^{K} \left[1 - \sigma(w^\top x_j)\right] , \tag{2}$$

where the bag $\mathbf{x} = \{x_j\}_{j=1}^K$ contains K examples.This model is sometimes referred to as the $noisy-OR$ and has been previously used by (Viola et al., 2006) in a boosting framework and (Maron & Lozano-Perez, 1998) in the Diverse Density algorithm. We use this model for Logistic Regression. A negative bag means that *all* examples in the bag are negative. Hence

$$p(y = 0|\mathbf{x}) = \prod_{j=1}^{K} \left[1 - \sigma(w^\top x_j)\right] . \tag{3}$$

5. Maximum Likelihood Estimator

Given the training data $\mathcal{D}$ the maximum likelihood (ML) estimate for w is given by

$$\widehat{w}_{\text{ML}} = \arg\max_w p(\mathcal{D}/w) = \arg\max_w \left[\log p(\mathcal{D}/w)\right] . \tag{4}$$

Define $p_i = p(y_i = 1|\mathbf{x}_i) = 1 - \prod_{j=1}^{K_i} \left[1 - \sigma(w^\top x_{ij})\right]$– the probability that the i^{th} bag $\mathbf{x}_i$ is positive. Assuming that the training bags are independent the log-likelihood can be written as

$$\log p(\mathcal{D}/w) = \sum_{i=1}^{N} y_i \log p_i + (1 - y_i) \log(1 - p_i). \tag{5}$$

A similar likelihood was also maximized by (Viola et al., 2006) using the AnyBoost framework, which views boosting as gradient descent in function space.

6. The MAP Estimator

The ML solution in practice can exhibit severe overfitting especially for high-dimensional data. This can be addressed by using a prior on w.

Prior We will assume zero mean Gaussian prior ($\mathcal{N}(w|0, \mathbf{A}^{-1})$) on the weights w with inverse covariance matrix $\mathbf{A} = diag(\alpha_1 \ldots \alpha_d)$ (also referred to as the *hyper-parameters*).

$$p(w) = (2\pi)^{-d/2} |\mathbf{A}^{-1}|^{-1/2} \exp\left(-\frac{w^\top \mathbf{A} w}{2}\right). \quad (6)$$

This encapsulates our prior belief that the individual weights in w are independent and close to zero with a variance parameter $1/\alpha_i$.

Posterior Once we observe the training data $\mathcal{D}$ we will update the prior to compute the posterior $p(w/\mathcal{D})$, which can be written as follows (using Bayes's rule)–$p(w/\mathcal{D}) = p(\mathcal{D}/w)p(w)/\int p(\mathcal{D}/w)p(w)dw$. This posterior can then be used to compute predictive distributions, which will typically involve high dimensional integrals. For computational efficiency we could use point estimates of w. In particular the *maximum a-posteriori* (MAP) estimate is given by

$$\widehat{w}_{\text{MAP}} \;=\; \arg\max_{w} \left[\log p(\mathcal{D}/w) + \log p(w)\right]. \quad (7)$$

Substituting for the log likelihood and the prior we have $\widehat{w}_{\text{MAP}} = \arg\max_w L(w)$, where

$$L(w) = \left[\sum_{i=1}^{N} y_i \log p_i + (1-y_i)\log(1-p_i)\right] - \frac{w^\top \mathbf{A} w}{2}. \quad (8)$$

Optimization Due to the non-linearity of the sigmoid we do not have a closed form solution and we have to use gradient based optimization methods. We use the Newton-Raphson update given by $w^{t+1} = w^t - \eta \mathbf{H}^{-1}\mathbf{g}$, where $\mathbf{g}$ is the gradient vector, $\mathbf{H}$ is the Hessian matrix, and η is the step length. The gradient is given by

$$\mathbf{g}(w) = \sum_{i=1}^{N} \left[y_i\beta_i - (1-y_i)\right] \sum_{j=1}^{K_i} x_{ij}\sigma(w^\top x_{ij}) - \mathbf{A}w, \quad (9)$$

where $\beta_i = \frac{1-p_i}{p_i}$. Note that $\beta_i = 1$ corresponds to the derivatives of the standard logistic regression updates. These term β_i can be thought of as the bag weight by which each instance weight gets modified. The Hessian matrix is given by

$$\mathbf{H}(w) = \sum_{i=1}^{N} \left[y_i\beta_i - (1-y_i)\right] \sum_{j=1}^{K_i} x_{ij}x_{ij}^T \sigma(w^\top x_{ij}) \sigma(-w^\top x_{ij}) - \sum_{i=1}^{N} y_i\beta_i(\beta_i+1) \left[\sum_{j=1}^{K_i} x_{ij}\sigma(w^\top x_{ij})\right] \left[\sum_{j=1}^{K_i} x_{ij}\sigma(w^\top x_{ij})\right]^T - \mathbf{A}. \quad (10)$$

Note that the Hessian matrix depends on the class labels also–unlike in regular logistic regression.

7. Bayesian MIL: Feature Selection

We imposed a prior of the form $p(w) = \mathcal{N}(w|0, \mathbf{A}^{-1})$, parameterized by d hyper-parameters $\mathbf{A} = diag(\alpha_1 \ldots \alpha_d)$. Clearly, as the precision $\alpha_k \to \infty$, *i.e*, the variance for w_k tends to zero (thus concentrating the prior sharply at zero). Hence, regardless of the evidence of the training data, the posterior for w_k will also be sharply concentrated on zero, thus that feature will not affect the classification result-hence, it is effectively removed out via feature selection. Therefore, the discrete optimization problem corresponding to feature selection (should each feature be included or not?), can be more easily solved via an easier continuous optimization over hyper-parameters . If one could maximize the marginal likelihood $p(\mathcal{D}|\mathbf{A})$ this would perform optimal feature selection. This approach is also known as the *type-II maximum likelihood* method in the Bayesian literature.

We choose the hyper-parameters to maximize the marginal likelihood.

$$\widehat{\mathbf{A}} = \arg\max_{\mathbf{A}} p(\mathcal{D}|\mathbf{A}) = \arg\max_{\mathbf{A}} \int p(\mathcal{D}|w)p(w|\mathbf{A})dw. \quad (11)$$

Since this integral is not easy to compute for our MIL model we use an approximation to the marginal likelihood via the Taylor series expansion. The marginal likelihood $p(\mathcal{D}|\mathbf{A})$ can be written as $p(\mathcal{D}|\mathbf{A}) = \int e^{\Psi(w)}dw$, where $\Psi(w) = \log p(\mathcal{D}|w) + \log p(w|\mathbf{A})$. Approximating Ψ using a second order Taylor series around $\widehat{w}_{\text{MAP}}$,

$$\Psi(w) \approx \Psi(\widehat{w}_{\text{MAP}}) + \frac{1}{2}(w - \widehat{w}_{\text{MAP}})\mathbf{H}(\widehat{w}_{\text{MAP}}, \mathbf{A})(w - \widehat{w}_{\text{MAP}})^\top. \quad (12)$$

Hence we have the following approximation to the

```
Input: $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i = \{x_{ij} \in \mathbb{R}^d\}_{j=1}^{K_i}$ is a
bag containing $K_i$ instances that share the same
label $y_i \in \{0,1\}$.
Output: A list of selected features and weight
vector $w$ for the linear classifier.
Initialize $\alpha_i = 1$ and $w_i = 0$ for $i = 1, \dots, d$.
repeat
    If $\alpha_i > \tau$ remove feature $w_i$.
    MAP estimate using the selected features.
    repeat
        Compute the gradient vector $\mathbf{g}$. (Eq. 9)
        Compute the Hessian matrix $\mathbf{H}$. (Eq. 10)
        Determine $\eta$ using a line search.
        Update $w \leftarrow w - \eta\mathbf{H}^{-1}\mathbf{g}$.
    until $\|\mathbf{g}\|/d < \epsilon_1$
    Update the hyper-parameters using
    $\alpha_i \leftarrow 1/(w_i^2 + \Sigma_{ii})$. (Eq. 17)
until $\max_i |\log\alpha_i^{curr} - \log\alpha_i^{prev}| < \epsilon_2$
In the experiments reported in this paper we use $\tau = 10^{12}$,
$\epsilon_1 = 10^{-5}$, and $\epsilon_2 = 10^{-3}$.
```

Algorithm 1: The proposed algorithm

marginal likelihood

$$\begin{aligned} & p(\mathcal{D}|\mathbf{A}) \\ \approx\ & e^{\Psi(\widehat{w}_{\rm MAP})}\int e^{\frac{1}{2}(w-\widehat{w}_{\rm MAP})\mathbf{H}(\widehat{w}_{\rm MAP},\mathbf{A})(w-\widehat{w}_{\rm MAP})^\top}dw \\ \approx\ & p(\mathcal{D}|\widehat{w}_{\rm MAP})p(\widehat{w}_{\rm MAP}|\mathbf{A})(2\pi)^{d/2}|-\mathbf{H}^{-1}(\widehat{w}_{\rm MAP},\mathbf{A})|^{1/2} \end{aligned} \tag{13}$$

Using the prior $p(w|\mathbf{A}) = \mathcal{N}(w|0, \mathbf{A}^{-1})$, the log marginal likelihood can be written as

$$\begin{aligned} \log p(\mathcal{D}|\mathbf{A}) \approx\ & \log p(\mathcal{D}|\widehat{w}_{\rm MAP}) - \frac{1}{2}\widehat{w}_{\rm MAP}^\top\mathbf{A}\widehat{w}_{\rm MAP} \\ +\ & \frac{1}{2}\log|\mathbf{A}| - \frac{1}{2}\log|-\mathbf{H}(\widehat{w}_{\rm MAP},\mathbf{A})|. \end{aligned} \tag{14}$$

The hyper-parameters $\mathbf{A}$ are found by maximizing this approximation to the log marginal likelihood. There is no closed-form solution for this. Hence we use a iterative re-estimation method by setting the first derivative to zero. The derivative can be written as

$$\begin{aligned} \frac{\partial\log p(\mathcal{D}|\mathbf{A})}{\partial\mathbf{A}} =\ & -\frac{1}{2}\widehat{w}_{\rm MAP}\widehat{w}_{\rm MAP}^\top + \frac{1}{2}\mathbf{A}^{-1} \\ -\ & \frac{1}{2}\mathbf{H}^{-1}(\widehat{w}_{\rm MAP},\mathbf{A}). \end{aligned} \tag{15}$$

Since $\mathbf{A} = diag(\alpha_1 \dots \alpha_d)$, we can further simplify

$$\frac{\partial\log p(\mathcal{D}|\mathbf{A})}{\partial\alpha_i} = -\frac{1}{2}\widehat{w}_i^2 + \frac{1}{2\alpha_i} - \frac{1}{2}\Sigma_{ii}, \tag{16}$$

where Σ_{ii} is the i^{th} diagonal element of $\mathbf{H}^{-1}(\widehat{w}_{\rm MAP},\mathbf{A})$. Assuming Σ_{ii} does not depend on $\mathbf{A}$ a simple update rule for the hyper-parameters can be written by equating the first derivative to zero.

$$\alpha_i^{\rm new} = \frac{1}{w_i^2 + \Sigma_{ii}}. \tag{17}$$

The final algorithm has two levels of iterations (see Algorithm 1): in an outer loop we update the hyper-parameters α_i and in an inner loop we find the MAP estimator $\widehat{w}_{\rm MAP}$ given the hyper-parameters. After a few iterations we find that the hyper-parameters–the inverse variances of the priors–for several features tend to infinity causing numerical problems in implementation. This means that those $w_i \to 0$ we can simply remove those irrelevant features from further consideration in future iterations. The proposed algorithm with feature selection can be considered as the extension of the Relevance Vector Machine (RVM) (Tipping, 2001) to multiple-instance learning framework.

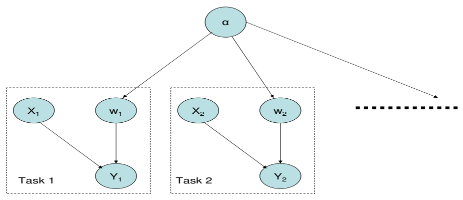

Figure 1. In multi-task learning tasks share the same prior.

8. Multi-task Learning

We are often faced with a shortage of training data for learning classifiers for a task. However we may have additional data for closely related, albeit non-identical tasks. For example in our CAD applications where we have to identify early stage cancers from CT scans, our data set includes images from CT scanners with two different reconstruction kernels–B50 and B60.

While training the classifier we could ignore this information and pool all the data together. However, there are some systematic differences that make the feature distributions slightly different. Alternatively, we could train a separate classifier for each kernel, but a large part of our data set is from one particular kernel (B60) and we have a smaller data set for the other (B50).

Here, we discuss another approach–*multi-task learning* (Caruana, 1997)– that tries to estimate models w^j for several classification tasks j in a joint manner. Multi-task learning can compensate for small sample size by using additional samples from related tasks, and exchanging statistical information between tasks. In a hierarchical Bayesian approach, the clas-

Table 1. Datasets used in our MIL experiments. d is the number of features.

Dataset	d	positive		negative	
		examples	bags	examples	bags
Musk1	166	207	47	269	45
Musk2	166	1017	39	5581	63
Elephant	230	762	100	629	100
Tiger	230	544	100	676	100

sifiers share a common prior $p(w^j|\mathbf{A})$ (See Figure 1). A separate classifier is trained for each task. However the optimal hyper-parameters of the shared prior are estimated from all the data sets simultaneously during the training. The update equation becomes (in place of Eq. 17): $\alpha_i^{\text{new}} = 1/\sum_{\text{task}j}(\widehat{w}_i^j)^2 + \Sigma_{ii}^j$.

9. Experimental Results

9.1. Datasets

Experiments were performed on four common benchmark data sets from the MIL literature (see Table 1).

Musk1 and Musk2 (Asuncion & Newman, 2007) The task is to predict whether a new drug molecule will bind to a target protein. However each molecule (bag) can have many different low energy shapes (instances) of which only one can actually bind with the target.

Elephant and Tiger The task is to search a repository to find images that contain objects of interest. An image is represented as a bag. An instance in a bag corresponds to a segment in the image; the object of interest is contained in at least one segment.

9.2. Competing Algorithms

Various learning algorithms have been adapted to the multiple learning scenario. We compare our proposed algorithm with a variant of Boosting, SVM, and Logistic Regression. Specifically we perform our experimental comparison for the following algorithms.

MI RVM The proposed multiple-instance algorithm with feature selection. This is completely automatic and does not require tuning any parameters.

RVM The proposed algorithm without multiple instance learning. This is same as MI RVM but every example is assigned to a unique bag.

MI The proposed multiple-instance algorithm without feature selection. We set $\mathbf{A} = \lambda\mathbf{I}$, where $\mathbf{I}$ is the identity matrix and λ is chosen by five-fold cross-validation.

MI Boost (Xin & Frank, 2004) This is a variant of the AdaBoost algorithm adapted for the multiple instance learning scenario. We boosted for 50-100 rounds.

Table 2. The AUC for different algorithms and datasets.

Set	MIRVM	RVM	MIBoost	MILR	MISVM	MI
Musk1	0.942	**0.951**	0.899	0.846	0.899	0.922
Musk2	**0.987**	0.985	0.964	0.795	-	0.982
Elephant	0.962	**0.979**	0.828	0.814	0.959	0.953
Tiger	**0.980**	0.970	0.890	0.890	0.945	0.956

Table 3. The average number of features selected per fold by different algorithms.

Dataset	Number of features	selected by RVM	selected by MI RVM	selected by MI Boost
Musk1	166	39	**14**	33
Musk2	166	90	**17**	32
Elephant	230	42	**16**	33
Tiger	230	56	**19**	37

MI SVM (Andrews et al., 2002) This is a bag-level SVM variant for MIL. We used the implementation publicly available at (Yang, 2006). We used a linear kernel and the regularization parameters was chosen by 5-fold cross-validation.

MI LR (Settles et al., 2008; Ray & Craven, 2005) This is a variant of Logistic Regression which uses the soft-max function to combine posterior probabilities over the instances of a bag. We used $\alpha = 3$ in the soft-max function. Non-linear conjugate gradient with tolerance set at 10^{-3} was used as the optimization routine.

Of all the above algorithms only our proposed method and the boosting one does automatic feature selection. We are not aware of any other multiple-instance algorithms which does automatic feature selection.

9.3. Evaluation Procedure

The results are shown for a 10-fold stratified cross-validation. The folds are split at the positive bag level, so that examples in the same positive bag will not be split. We plot the Receiver Operating Characteristics (ROC) curve for various algorithms (see Figure 2). The ROC curve is a plot of False Positive Rate (FPR) vs True Positive Rate (TPR) as the decision threshold of the classifier θ is varied from ∞ to $-\infty$. The TPR is computed on a bag level–*i.e.*, a bag is predicted as positive if at least one on the instances in classified as positive. The ROC curve is plotted by pooling the prediction of the algorithm across all folds as in (Ray & Craven, 2005). We also report the area under the ROC curve (AUC) in Table 2.

9.4. Results

Comparison with other methods. From Figure 2 and Table 2 we see that among other MIL algorithms the ROC for the proposed method clearly dominates

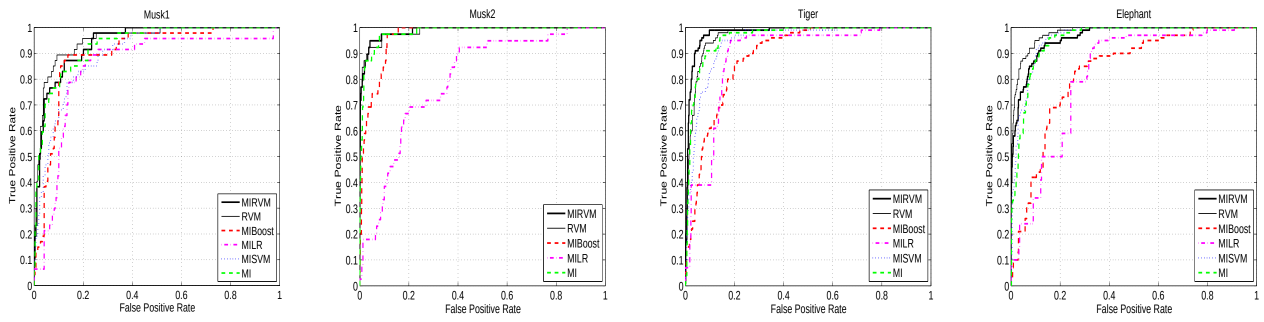

Figure 2. The ROC Curves for the different data sets and the different algorithms.

the other methods. However it is interesting to note that plain RVM is better than MI RVM for Musk 1 and Elephant data sets. This confirms the surprising observation in (Ray & Craven, 2005) that for some MIL benchmarks standard supervised learning algorithm may be more accurate than MIL algorithms.

Number of features selected. Table 3 compares the number of features selected by MI RVM, RVM, and the MI Boost algorithm. It can be seen that the proposed MI RVM algorithm selects the least number of features. Selecting features in a multiple instance setting reduces the number of features selected by half.

Does feature selection help? From Table 2 we see that the AUC with feature selection is higher than that without feature selection. Thus we are able to achieve better performance and at the same time use a smaller set of features.

Runtime The proposed algorithm and the MI Boost are orders of magnitude faster than other MIL methods. As a result MI SVM and MI LR could not be run on our CAD experiments described in the next section. Also the proposed method has no free parameters to tune. The runtime of our algorithm scales as $\mathcal{O}(d^3)$ with the number of features. This is because we need to compute the inverse of the $d \times d$ Hessian matrix.

10. Computer Aided Diagnosis

In computer aided diagnosis (CAD) the goal is to detect potentially malignant nodules, tumors, emboli, or lesions in medical images like computed tomography (CT), X-ray, MRI etc. A CAD system aids the radiologist by marking the location of likely anomaly on a medical image. Figure 3 shows two pulmonary emboli (PE) in a CT scan. PE (blood clots in the lung), is a potentially life-threatening condition. An early and accurate diagnosis is the key to survival. Computed tomography angiography (CTA) has emerged as an accurate diagnostic tool.

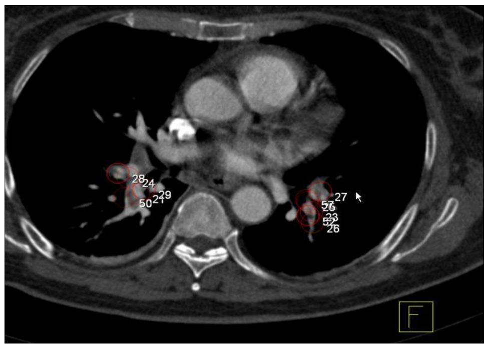

Figure 3. Sample pulmonary emboli in a Lung CT scan along with the candidates which point to it.

Most CAD systems consist of the following three steps–(1) *Candidate generation*–this step identifies potentially unhealthy regions of interest. While this step can detect most of the anomalies, the number of false positives will be extremely high (60-100 false positives/ patient). (2) *Feature computation*–computation of a set of descriptive morphological features for each of the candidates. (3) *Classification*–labeling of each candidate as a nodule or not by a classifier. The goal of the classifier is to reduce the number of false positives without appreciable decrease in the sensitivity.

In order to train a classifier, a set of CT scans is collected from hospitals. These scans are then read by expert radiologists who mark the pulmonary emboli locations–this constitutes our ground truth for learning. The candidate generation step generates a lot of potential candidates. Any candidate which is close to the radiologist mark (for example within a certain distance) is considered a positive example for training and the rest of the candidates are considered as negative examples. Based on a set of features computed for these candidates we intend to train a classifier.

Table 4. Datasets used in our PE CAD MIL experiments.

Dataset	Features	positive		negative
		examples	bags	examples
Training	134	514	312	4619
Validation	134	305	214	3246

10.1. Multiple Instance Learning for CAD

The candidate generation step very often produces a lot of candidates which are spatially close to each other (See Figure 3 for two PE appearing in a CT scan). All these candidates point to the same ground truth and can considered to share the same label for training. A single instance classifier can be trained using the labeled candidates. In this work we use the multiple instance learning algorithm by recognizing the fact that all candidates which point to the same radiologist mark can be considered as a positive bag (Fung et al., 2007). There is another important reason why MIL is a natural framework for CAD. The candidate generation algorithm produces a lot of spatially close candidates. Even if one of these is highlighted to the radiologist and other adjacent or overlapping candidates are missed, the underlying embolism would still have been detected. Hence while evaluating the performance of CAD systems we use the bag level sensitivity, *i.e.*, a classifier is successful in detecting an embolism if at least one of the candidates pointing to it is predicted as a PE. MIL naturally lends itself to model our desired accuracy measure during training itself.

Another important requirement is that run time of the classifier during testing should be as small as possible. The candidate generation step generally produces thousands of candidates for a CT scan. Computing all the features can be very time-consuming. Hence it is imperative that the final classifier uses as few features as possible without any decrease in the sensitivity. The proposed classifier automatically selects features for multiple-instance classification.

10.2. Experiments

Table 4 summarizes the PE CAD data sets we use in our experiments. Note that unlike the previous four data sets we do not have negative bags. Every negative example is considered a negative bag. The classifier is trained on the training set and tested on a separate validation set. Since we are interested in the number of False Positives per volume (patient) we plot Free response ROC (FROC) curves for the validation set (see Figure 4). MI RVM gives a substantial improvement over the single instance RVM approach and also the MI Boost method. The MI SVM and MI LR could not be run for the large CAD data set. MI RVM algorithm selected 21 features in contrast to the 34 features selected by the single instance RVM algorithm

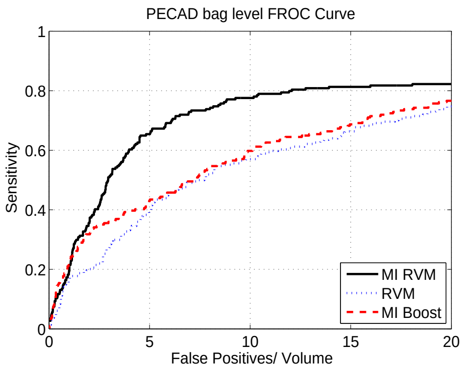

Figure 4. The bag level FROC curve for the PECAD validation set.

10.3. Multi-task Learning Experiments

Lung cancer is a leading cause of cancer related death in western countries. However early detection can substantially improve survival. Automatic CAD systems can be developed to identify suspicious regions such as solid nodules or ground-glass opacities (GGO) in CT scans of the lung. A solid nodule is defined as an area of increased opacity more than 5mm in diameter which completely obscures underlying vascular marking. Ground-glass opacity(GGO) is defined as an area of a slight, homogenous increase in density, which did not obscure underlying bronchial and vascular markings. Figure 5 shows an example nodule and GGO.

Detecting nodules and GGOs are two closely related tasks although each has its own respective characteristics. Hence multi-task learning is likely to be beneficial, even when building a specific model for each task. To train such a system we used 15 CT scans which included GGOs and 23 CT scans that included nodules. The model accuracy was validated on a held out set of 86 CT scans that included nodules. Figure 6 compares the FROC curves for nodule detection system designed in two different ways. (1) Single task learning: the classifier was learnt only using nodule data. (2) Multi-task learning: the classifier was learnt using nodule data and GGO data. As Figure 6 shows, inductive transfer using the proposed scheme the improves accuracy of the multiple instance learning system, when we have a limited amount of training data.

11. Conclusion

In this paper we proposed a novel MIL algorithm that automatically selects the features relevant for *multi-*

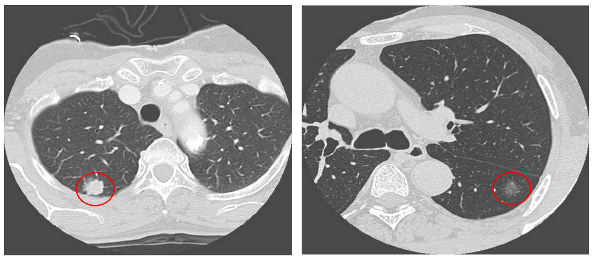

Figure 5. Lung CT image showing a sample (a)nodule and (b) GGO. Figure reprinted from (Suzuki et al., 2006).

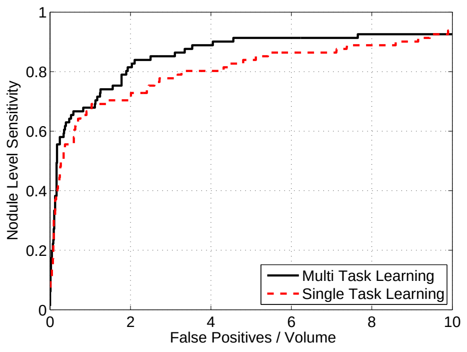

Figure 6. Multi-task learning experiments The bag level FROC curve for the validation set.

ple instance classification. The proposed algorithm is more accurate than other competing MIL methods, both on benchmark data sets and on real life CAD problems. Our experiments also validate the previous observation of (Ray & Craven, 2005) that on some multiple instance benchmarks the single instance classifier is slightly more accurate. For all domains, the number of features selected by our algorithm is much smaller than that for the corresponding single instance classifier. Inductive transfer improves accuracy in data poor CAD applications.

Acknowledgements

We would like to thank Sangmin Park, Vikram Anand, Anna Jerebko, and Marcos Salgonicoff for help with the CAD data sets. We would also like to thank the reviewers for their suggestions which helped to improve the overall quality of the paper.

References

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2002). Support vector machines for multiple-instance learning. In *Advances in neural information processing systems 15*.

Asuncion, A., & Newman, D. J. (2007). UCI machine learning repository, `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Caruana, R. (1997). Multi-task learning. *Machine Learning*, *28*, 41–75.

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

Fung, G., Dundar, M., Krishnapuram, B., & Rao, R. B. (2007). Multiple instance learning for computer aided diagnosis. In *Advances in neural information processing systems 19*, 425–432.

Maron, O., & Lozano-Perez, T. (1998). A framework for multiple-instance learning. In *Advances in neural information processing systems 10*, 570–576.

Ray, S., & Craven, M. (2005). Supervised versus multiple instance learning: an empirical comparison. *ICML '05: Proceedings of the 22nd international conference on Machine learning* (pp. 697–704).

Settles, B., Craven, M., & Ray, S. (2008). Multiple-instance active learning. In *Advances in neural information processing systems 20*.

Suzuki, K., Kusumoto, M., Watanabe, S.-i., Tsuchiya, R., & Asamura, H. (2006). Radiologic Classification of Small Adenocarcinoma of the Lung: Radiologic-Pathologic Correlation and Its Prognostic Impact. *Ann Thorac Surg*, *81*, 413–419.

Tipping, M. E. (2001). Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, *1*, 211–244.

Viola, P., Platt, J., & Zhang, C. (2006). Multiple instance boosting for object detection. In *Advances in neural information processing systems 18*, 1417–1424.

Wang, J., & Zucker, J. (2000). Solving the multiple-instance problem: A lazy learning approach. In *Proceedings of the 17th international conference on machine learning*, 1119–1125.

Xin, X., & Frank, E. (2004). Logistic regression and boosting for labeled bags of instances. *Proc 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 272–281).

Yang, J. (2006). MILL: A multiple instance learning library. `http://www.cs.cmu.edu/~juny/MILL`.