A Regularization Framework for Multiple-Instance Learning

Pak-Ming Cheung PAKMING@CS.UST.HK
James T. Kwok JAMESK@CS.UST.HK
Department of Computer Science, Hong Kong University of Science and Technology, Kowloon, Hong Kong

Abstract

This paper focuses on kernel methods for multi-instance learning. Existing methods require the prediction of the bag to be identical to the maximum of those of its individual instances. However, this is too restrictive as only the sign is important in classification. In this paper, we provide a more complete regularization framework for MI learning by allowing the use of different loss functions between the outputs of a bag and its associated instances. This is especially important as we generalize this for multi-instance regression. Moreover, both bag and instance information can now be directly used in the optimization. Instead of using heuristics to solve the resultant nonlinear optimization problem, we use the constrained concave-convex procedure which has well-studied convergence properties. Experiments on both classification and regression data sets show that the proposed method leads to improved performance.

1. Introduction

Multi-instance (MI) learning was first introduced by Dietterich et al. (1997) in the context of drug activity prediction. The task is to predict whether a drug molecule can bind to the targets (enzymes or cell-surface receptors). Each drug molecule (called a *bag*) can have multiple low-energy shapes or conformations (*instances*). The molecule is considered useful as a drug if one of its conformations can bind to the targets. However, biochemical data can only tell the binding capability of a molecule, but not a particular conformation. Thus, while each training pattern has a known label in supervised learning, only the bags (but not the individual instances) have known labels in MI learning. In other words, MI learning only provides weak label information of the training data.

Following the seminal work of Dietterich et al. (1997), a number of new MI learning methods emerged. Examples include the Diverse Density (DD) (Maron & Lozano-Perez, 1998), EM-DD (Zhang & Goldman, 2002), Citation-kNN (Wang & Zucker, 2000), MI SVMs (Andrews et al., 2003) and generalized MI learning (Tao et al., 2004). Besides classification, progress has also been made on MI learning with real-valued outputs (Ray & Page, 2001; Amar et al., 2001; Dooly et al., 2002). Moreover, many applications are now considered as MI problems. Examples include content-based image retrieval (Chen & Wang, 2004; Maron & Ratan, 1998), where each image is a bag and each local image patch an instance, and computer hard-drive failure prediction (Murray et al., 2005).

In this paper, we focus on kernel methods, which have been highly successful in various machine learning problems (Schölkopf & Smola, 2002). To adapt to MI learning, one common approach is to design kernels for bags. A standard support vector machine (SVM) can then be used with these so-called MI kernels (Chen & Wang, 2004; Gärtner et al., 2002). Note that the underlying quadratic programming (QP) problem only involves dual variables corresponding to the bags, but not instances. A more direct approach that takes the instances into account is to associate the dual variables with the instances, but not the bags (Andrews et al., 2003). Although the bags are now no longer involved in the optimization, the bag label information of a particular bag B_i of instances $\{\mathbf{x}_{ij}\}_{j=1}^{n_i}$ is still used implicitly by requiring

$$f(B_i) = \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}), \tag{1}$$

where f is the function to be learned by the SVM. Moreover, instead of having a QP, the resultant optimization problem is a mixed-integer problem. Andrews et al. (2003) proposed a simple optimization heuristic for solving this problem. However, unlike

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

QPs which have been well-studied, the convergence properties of this heuristic are unclear.

Besides, as only the sign is important in classification, namely that $\text{sign}(f(B_i)) = \text{sign}(\max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}))$, so (1) may be too restrictive. Moreover, when the bag labels are noisy, a mislabeled $f(B_i)$ will incorrectly constrain the predictions on its constituent instances via the condition (1), and consequently wrongly biases the learning of f. As will be discussed in Section 4, this issue becomes even more important in MI regression.

In this paper, we thus relax condition (1) by introducing a loss function between $f(B_i)$ and the associated $f(\mathbf{x}_{ij})$'s. This allows both the bags and instances to directly participate in the optimization process, and the learned function to be smooth over both bags and instances. Moreover, instead of using optimization heuristics as in (Andrews et al., 2003), we will use the disciplined *constrained concave-convex procedure* (Smola et al., 2005) to perform the optimization.

The rest of this paper is organized as follows. Section 2 gives a brief review on the concave-convex procedure. Section 3 then describes the proposed formulation for MI classification, together with a similar extension for MI regression in Section 4. Experimental results are presented in Section 5, and the last section gives some concluding remarks.

2. Concave-Convex Procedure

The *concave-convex procedure* (Yuille & Rangarajan, 2003) is an optimization tool for problems whose objective function can be expressed as a difference of convex functions. In the optimization literature, it is often known as the *difference of convex functions* algorithm (DCA) (Pham Dinh & Hoai An, 1998). While Yuille and Rangarajan (2003) considered only linear constraints, Smola et al. (2005) generalized this to the *constrained concave-convex procedure* (CCCP) for problems with a concave-convex objective function with concave-convex constraints.

Consider the following optimization problem:

$$\begin{array}{ll} \min_{\mathbf{x}} & f_0(\mathbf{x}) - g_0(\mathbf{x}) \\ \text{s.t.} & f_i(\mathbf{x}) - g_i(\mathbf{x}) \le c_i, \;\; i = 1, \ldots, m, \end{array} \tag{2}$$

where f_i, g_i $(i = 0, \ldots, m)$ are real-valued, convex and differentiable functions on $\mathbb{R}^n$, and $c_i \in \mathbb{R}$. Given an initial $\mathbf{x}^{(0)}$, CCCP computes $\mathbf{x}^{(t+1)}$ from $\mathbf{x}^{(t)}$ by replacing $g_i(\mathbf{x})$ with its first-order Taylor expansion at $\mathbf{x}^{(t)}$, and then setting $\mathbf{x}^{(t+1)}$ to the solution of the relaxed optimization problem:

$$\begin{array}{ll} \min\limits_{\mathbf{x}} & f_0(\mathbf{x}) - \left[g_0(\mathbf{x}^{(t)}) + \nabla g_0(\mathbf{x}^{(t)})'(\mathbf{x} - \mathbf{x}^{(t)}) \right] \\ \text{s.t.} & f_i(\mathbf{x}) - \left[g_i(\mathbf{x}^{(t)}) + \nabla g_i(\mathbf{x}^{(t)})'(\mathbf{x} - \mathbf{x}^{(t)}) \right] \le c_i, \end{array} \tag{3}$$

Here, the superscript $'$ denotes the vector transpose, and $\nabla g(\hat{\mathbf{x}})$ is the gradient of the function g at $\hat{\mathbf{x}}$. It can be shown that CCCP converges to a local minimum solution of (2) (Smola et al., 2005). For non-smooth functions, it can be easily shown that the gradient should then be replaced by the subgradient.

3. Multi-Instance Classification

In MI classification, we are given a set of training bags $\{(B_1, y_1), \ldots, (B_m, y_m)\}$, where $B_i = \{\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{in_i}\}$ is the ith bag containing instances $\mathbf{x}_{ij}$'s, and $y_i \in \{\pm 1\}$. As mentioned in Section 1, we take both bags and instances directly into account. Hence, we define a general loss function that depends on both the training bags and training instances:

$$V\left(\{B_i, y_i, f(B_i)\}_{i=1}^{m}, \{f(\mathbf{x}_{ij})\}_{j=1\ i=1}^{n_i\ \ m}\right). \tag{4}$$

3.1. The Loss Function V

In this paper, we split the loss function V in (4) into two parts. The first part considers the loss between each bag label y_i and its corresponding prediction $f(B_i)$. Here, as in the SVM, we use the hinge loss $(1 - y_i f(B_i))_+$ where $(z)_+ = \max(0, z)$. The second part considers the loss between the prediction of each bag $f(B_i)$ and those of its constituent instances $\{f(\mathbf{x}_{ij}) \mid j = 1, \ldots, n_i\}$. In general, this can be defined in various ways. As mentioned in Section 1, (1) is often enforced, which is the same as using the loss $\ell(f(B_i), \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}))$, where

$$\ell(v_1, v_2) = \begin{cases} 0 & \text{if } v_1 = v_2, \\ \infty & \text{otherwise.} \end{cases} \tag{5}$$

To relax condition (1), we will consider some other popular loss functions, including the L1 loss:

$$\ell(v_1, v_2) = |v_1 - v_2|, \tag{6}$$

the L2 loss:

$$\ell(v_1, v_2) = (v_1 - v_2)^2, \tag{7}$$

and the ϵ-insensitive loss

$$\ell(v_1, v_2; \epsilon) = \max\{0, |v_1 - v_2| - \epsilon\}. \tag{8}$$

Putting the two parts together, we have

$$V\left(\{B_i, y_i, f(B_i)\}_i, \{f(\mathbf{x}_{ij})\}_{ij}\right) \quad (9)$$
$$= \frac{1}{m}\sum_{i=1}^{m}(1 - y_i f(B_i))_+ + \frac{\lambda}{m}\sum_{i=1}^{m}\ell(f(B_i), \max_j f(\mathbf{x}_{ij})),$$

where λ is a parameter that trades off the two components.

On the other hand, if we ignore the loss between the prediction of each bag $f(B_i)$ and those of its constituent instances $\{f(\mathbf{x}_{ij})\}$ (or, equivalently, by setting λ in (9) to zero), then the loss function $V(\cdot)$ in (9) reduces to $\frac{1}{m}\sum_{i=1}^{m}(1 - y_i f(B_i))_+$. This is then the same as using the standard SVM with the multi-instance kernel (Gärtner et al., 2002).

3.2. Representer Theorem

The representer theorem plays a central role in the development of the SVM, as it allows the optimization in a possibly infinite-dimensional space to be reduced to a finite-dimensional optimization problem. However, as the proposed optimization problem now involves variables coming from both the bags and instances, it appears that the representer theorem no longer holds in our MI setting.

Indeed, this can be resolved by the simple observation that an instance can also be considered as a bag of size one. Let $\mathcal{F}$ be the space of instances and $2^{\mathcal{F}}$ the power set of instances. Let $\mathcal{H}$ be a RKHS of functions[1] $f : 2^{\mathcal{F}} \to \mathbb{R}$, with associated kernel function k. For any $\mathbf{x} \in \mathcal{F}$, $\{\mathbf{x}\} \in 2^{\mathcal{F}}$, and so $f(\{\mathbf{x}\})$ is well-defined for $f \in \mathcal{H}$. With an abuse of notation, we simply write $f(\{\mathbf{x}\})$ as $f(\mathbf{x})$, and $k(\{\mathbf{x}\}, \cdot)$ as $k(\mathbf{x}, \cdot)$.

Denote the RKHS norm of $\mathcal{H}$ by $\|f\|_{\mathcal{H}}$ and $\Omega : [0, \infty) \to \mathbb{R}$ a strictly monotonically increasing function. By the representer theorem, each minimizer $f \in \mathcal{H}$ of the regularized risk

$$\gamma V\left(\{B_i, y_i, f(B_i)\}_{i=1}^m, \{f(\mathbf{x}_{ij})\}_{j=1\ i=1}^{n_i\ m}\right) + \frac{1}{2}\Omega\left(\|f\|_{\mathcal{H}}^2\right)$$

admits a representation of the form

$$f(\mathbf{x}) = \sum_{i=1}^{m}\left(\alpha_{i0}k(\mathbf{x}, B_i) + \sum_{j=1}^{n_i}\alpha_{ij}k(\mathbf{x}, \mathbf{x}_{ij})\right), \quad (10)$$

where all $\alpha_{i0}, \alpha_{ij} \in \mathbb{R}$. For simplicity, denote $n = \sum_{i=1}^{m} n_i$ and $\boldsymbol{\alpha}$ the $(m+n)$-dimensional vector consisting of all the α_{i0}'s and α_{ij}'s.

[1]Note that this is also the RKHS implicitly used with the MI kernel (Gärtner et al., 2002).

3.3. Optimization using CCCP

Using $\Omega\left(\|f\|_{\mathcal{H}}^2\right) = \|f\|_{\mathcal{H}}^2$ as in the SVM, and the loss function as discussed in Section 3.1, we have the following optimization problem:

$$\begin{aligned}\min_{f\in\mathcal{H},\boldsymbol{\xi}} \quad & \frac{1}{2}\|f\|_{\mathcal{H}}^2 + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} \\ & + \frac{\gamma\lambda}{m}\sum_{i=1}^{m}\ell(f(B_i), \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij})) \\ \text{s.t.} \quad & y_i f(B_i) \geq 1 - \xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0},\end{aligned} \quad (11)$$

where $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_m]'$ are slack variables for the errors on the training bags, $\mathbf{0} = [0, \ldots, 0]'$ and $\mathbf{1} = [1, \ldots, 1]'$ are vectors of 0's and 1's, respectively, and γ is a user-defined parameter that trades off model complexity with the error.

3.3.1. Using the L1 Loss for $\ell(\cdot, \cdot)$

In this Section, we first consider the use of the L1 loss (6) for $\ell(v_1, v_2)$. We can then rewrite (11) as:

$$\begin{aligned}\min_{f\in\mathcal{H},\boldsymbol{\xi},\boldsymbol{\delta}} \quad & \frac{1}{2}\|f\|_{\mathcal{H}}^2 + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & y_i f(B_i) \geq 1 - \xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & -\delta_i \leq f(B_i) - \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}) \leq \delta_i,\end{aligned}$$

where $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_m]'$. This is also equivalent to

$$\begin{aligned}\min_{f\in\mathcal{H},\boldsymbol{\xi},\boldsymbol{\delta}} \quad & \frac{1}{2}\|f\|_{\mathcal{H}}^2 + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & y_i f(B_i) \geq 1 - \xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & f(\mathbf{x}_{ij}) - \delta_i \leq f(B_i), \\ & f(B_i) - \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}) \leq \delta_i.\end{aligned} \quad (12)$$

Without loss of generality, assume that the bags and instances are ordered in the order $(B_1, \ldots, B_m, \mathbf{x}_{11}, \ldots, \mathbf{x}_{1n_1}, \ldots, \mathbf{x}_{m1}, \ldots, \mathbf{x}_{mn_m})$. Each object (bag or instance) in the training set can then be indexed by the following function $\mathcal{I}$:

$$\mathcal{I}(B_i) = i, \quad \mathcal{I}(\mathbf{x}_{ij}) = m + \sum_{k=1}^{i-1} n_k + j,$$

for $j = 1, \ldots, n_i$ and $i = 1, \ldots, m$. With this ordering, one obtains the $(m+n) \times (m+n)$ kernel matrix $\mathbf{K}$ defined on all objects in the training set. Denote the ith column of $\mathbf{K}$ by $\mathbf{k}_i$.

Using the representer theorem, we have $f(B_i) = \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}+b$ and $f(\mathbf{x}_{ij}) = \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}+b$ from (10). Here, we have also included an optional bias b. Moreover, $\|f\|^2_{\mathcal{H}} = \boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha}$. The optimization problem in (12) then becomes:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\delta},b} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & y_i(\mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}+b) \geq 1-\xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} - \delta_i \leq \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}, \\ & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}) \leq \delta_i. \end{aligned} \tag{13}$$

Note that the objective function is quadratic and the first three constraints are linear w.r.t. the variables. The last constraint (13) is, though nonlinear, a difference between two convex functions. Hence, we can solve this with the constrained concave-convex procedure (CCCP) in Section 2.

Notice that while $\max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha})$ in (13) is convex, it is a non-smooth function of $\boldsymbol{\alpha}$. To use the CCCP, we have to replace the gradients by the subgradients. Now, for the pointwise maximum function $h(\mathbf{x}) = \max_{1\leq i\leq p} h_i(\mathbf{x})$, its subdifferential[2] at $\mathbf{x}_0$ is the convex hull of the union of subgradients of "active" functions at $\mathbf{x}_0$, i.e., $\partial h(\mathbf{x}_0) = \mathrm{conv}\big(\cup_{i\in S(\mathbf{x}_0)}\partial h_i(\mathbf{x}_0)\big)$, where conv denotes the convex hull, and $S(\mathbf{x}_0)$ is the nonempty index set of all $i \in \{1,\ldots,p\}$ for which $h_i(\mathbf{x}_0) = h(\mathbf{x}_0)$. Thus, we obtain that for $i = 1,\ldots,m$, $\partial \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}) = \Big\{\sum_{j=1}^{n_i}\beta_{ij}\mathbf{k}_{\mathcal{I}(\mathbf{x}_{ij})} \mid \beta_{ij}\in\mathbb{R}^+\Big\}$, where

$$\beta_{ij}\begin{cases} = 0, & \text{if } \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} \neq \max_{k=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ik})}\boldsymbol{\alpha}), \\ \geq 0, & \text{otherwise,}\end{cases} \tag{14}$$

with

$$\sum_{j=1}^{n_i}\beta_{ij} = 1. \tag{15}$$

At the tth iteration, denote the current $\boldsymbol{\alpha}$ estimate and the corresponding β_{ij} by $\boldsymbol{\alpha}^{(t)}$ and $\beta_{ij}^{(t)}$, respectively. CCCP then replaces $\max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha})$ in the constraint by

$$\max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}^{(t)}) + \sum_{j=1}^{n_i}\beta_{ij}^{(t)}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}(\boldsymbol{\alpha}-\boldsymbol{\alpha}^{(t)}). \tag{16}$$

[2] A subgradient of f at $\mathbf{x}$ is any vector $\mathbf{g}$ that satisfies the inequality $f(\mathbf{y}) \geq f(\mathbf{x}) + \mathbf{g}'(\mathbf{y}-\mathbf{x})$ for all $\mathbf{y}$. The subdifferential of f at $\mathbf{x}$ is the set of all subgradients of f at $\mathbf{x}$.

Note from (14) that $\beta_{ij} \neq 0$ only for the active $\mathbf{x}_{ij}$'s. Hence,

$$\begin{aligned}\sum_{j=1}^{n_i}\beta_{ij}^{(t)}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}^{(t)} &= \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}^{(t)})\sum_{\beta_{ij}\neq 0}\beta_{ij}^{(t)} \\ &= \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}^{(t)}),\end{aligned}$$

on using (15), and (16) reduces to $\sum_{j=1}^{n_i}\beta_{ij}^{(t)}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}$. The optimization problem (3) in our case is then:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\delta},b} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & y_i(\mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}+b) \geq 1-\xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} - \delta_i \leq \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}, \\ & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - \sum_{j=1}^{n_i}\beta_{ij}^{(t)}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} \leq \delta_i, \end{aligned}$$

which is a standard QP problem. Following CCCP, the obtained $\boldsymbol{\alpha}$ solution from this QP is then used as $\boldsymbol{\alpha}^{(t+1)}$ and the iteration continues until convergence.

In the experiments, we initialize $\beta_{ij}^{(0)} = 1/n_i$ for $i = 1,\ldots,m$. Moreover, recall that the subgradient need not be unique. For convenience, we pick the subgradient with

$$\beta_{ij} = \begin{cases} 0, & \text{if } \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} \neq \max_{k=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ik})}\boldsymbol{\alpha}), \\ 1/n_a, & \text{otherwise,}\end{cases}$$

where n_a is the number of active $\mathbf{x}_{ij}$'s. Note that any other choices may also be used.

3.3.2. Using the Loss Function in (Andrews et al., 2003)

Recall that if we enforce (1) (which is the same as using (5) for $\ell(\cdot,\cdot)$), then the loss function $V(\cdot)$ in (9) becomes $\frac{1}{m}\sum_{i=1}^{m}(1 - y_i\max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}))_+$, which is the same as that used for the MI-SVM in (Andrews et al., 2003). Moreover, as $V(\cdot)$ does not depend on the $f(B_i)$'s, it can be easily seen from the proof of the representer theorem in (Schölkopf & Smola, 2002) that all the α_{i0}'s in (10) are zero. Therefore, the obtained decision function in (10) is only spanned by a subset of the training instances, and is thus the same as that in (Andrews et al., 2003).

If we use CCCP, we will obtain from (11) the following

optimization problem:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\xi},b} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} \\ \text{s.t.} \quad & \text{if } y_i = +1,\ \left(\sum_{j=1}^{n_i}\beta_{ij}^{(t)}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} + b\right) \geq 1 - \xi_i, \\ & \text{if } y_i = -1,\ -(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} + b) \geq 1 - \xi_i \\ & \boldsymbol{\xi} \geq \mathbf{0}.\end{aligned}$$

By picking the subgradient with

$$\beta_{ij} = \begin{cases} 1, & \text{if } j = \arg\max_{k=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ik})}\boldsymbol{\alpha}), \\ 0, & \text{otherwise,}\end{cases}$$

this becomes identical to the optimization heuristic for the MI-SVM used in (Andrews et al., 2003) and β_{ij} plays the role of the "selector variable" there. In other words, Andrews et al. (2003) is indeed using the CCCP with a particular choice of the subgradient. This also explains why they can "achieve competitive results even with the simpler optimization heuristics", as Pham Dinh and Hoai An (1998) stated that CCCP (which is called DCA there) can often converge to a global solution. Moreover, Andrews et al. (2003) only considers this as a heuristic and thus no convergence proof is given. On the contrary, we know that CCCP, using any of the loss functions discussed in this Section, is actually guaranteed to decrease the objective function monotonically and converge to a local minimum (Smola et al., 2005). As illustrated in (Yuille & Rangarajan, 2003), this provides another example that CCCP can be used to understand and prove the convergence of existing optimization algorithms.

3.3.3. Using Other Loss Functions for $\ell(\cdot,\cdot)$

Other loss functions can be used for $\ell(\cdot,\cdot)$. Using the same notations as in Section 3.3.1, the optimization problem with the L2 loss (7) is:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\delta},b} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\boldsymbol{\delta} \\ \text{s.t.} \quad & y_i(\mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} + b) \geq 1 - \xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} - \delta_i \leq \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}, \\ & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}) + \delta_i \leq 0,\end{aligned}$$

while that with the ϵ-insensitive loss (8) is:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\delta},\boldsymbol{\delta}^*,b} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \frac{\gamma}{m}\boldsymbol{\xi}'\mathbf{1} + \frac{\gamma\lambda}{m}(\boldsymbol{\delta}'\mathbf{1} + \boldsymbol{\delta}^{*\prime}\mathbf{1}) \\ \text{s.t.} \quad & y_i(\mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} + b) \geq 1 - \xi_i, \\ & \boldsymbol{\xi} \geq \mathbf{0}, \\ & \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} - \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} \leq \epsilon + \delta_i, \\ & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - \max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}) \leq \epsilon + \delta_i^*.\end{aligned}$$

Here, $\boldsymbol{\delta}^* = [\delta_1^*, \ldots, \delta_m^*]'$ is another vector of slack variables.

CCCP can again be straight-forwardly applied. It can be easily shown that the net effect is to replace $\max_{j=1,\ldots,n_i}(\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha})$ in the last constraint by $\sum_{j=1}^{n_i}\beta_{ij}\mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha}$ as in Section 3.3.1, and the resultant relaxed optimization problem is then a QP problem.

4. Multi-Instance Regression

In this Section, we show how the same regularization framework can be easily adapted for MI regression.

4.1. The Loss Function

As in Section 3, we split the loss function V into two parts. The first part considers the loss between the value of each bag and its corresponding prediction. As in ν-support vector regression (Schölkopf & Smola, 2002), we use the ϵ-insensitive loss and an extra $\nu\epsilon$ term (where ν is a user-defined parameter) to penalize the value of ϵ.

The second part considers the loss between the prediction of each bag and those of its constituent instances. Its design is, however, more complicated than that in classification (Section 3.1). Recall that in classification, the assumption is that a bag is positive if at least one of its constituent instances is positive. Hence, we can simply measure the difference between $f(B_i)$ and $\max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij})$ as in Section 3.1. However, in MI regression, it is less certain how the bag output should be related to the outputs of its constituent instances.

Ray and Page (2001) assumed that there is one *primary instance* in each bag that is responsible for the output of the bag. However, as it is not known which instance in the bag is the primary instance, this problem can be shown to be NP-complete, even when only a linear regression model is used (Ray & Page, 2001).

In this paper, we make the simplifying assumption that the primary instance is the one with the highest output value. Using the loss function[3] in Section 3.1,

[3]For simplicity, we only consider the L1 loss for ℓ here.

and introducing slack variables $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_m]', \boldsymbol{\xi} = [\xi_1, \ldots, \xi_m]'$ and $\boldsymbol{\xi}^* = [\xi_1^*, \ldots, \xi_m^*]'$, we obtain the following optimization problem:

$$\begin{aligned} \min_{f \in \mathcal{H}, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \epsilon, \boldsymbol{\delta}} \quad & \frac{1}{2}\|f\|_{\mathcal{H}}^2 + \gamma\nu\epsilon + \frac{\gamma}{m}(\boldsymbol{\xi}'\mathbf{1} + \boldsymbol{\xi}^{*\prime}\mathbf{1}) + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & f(B_i) - y_i \le \epsilon + \xi_i, \\ & y_i - f(B_i) \le \epsilon + \xi_i^*, \\ & -\delta_i \le f(B_i) - \max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij}) \le \delta_i, \quad (17) \\ & \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\delta} \ge \mathbf{0}, \ \epsilon \ge 0. \end{aligned}$$

Note that our assumption used in selecting the primary instance is valid in some applications, such as on the problem of predicting the strengths of synthetic musk molecules in (Dooly et al., 2002). While it may not be always correct, some error can still be tolerated with the use of the slack variable δ_i in (17). On the contrary, enforcing the hard constraint in (1) is certainly undesirable in this regression setting.

4.2. Optimization Problem in CCCP Iteration

Using the representer theorem and the CCCP as in Section 3, we obtain the following relaxed optimization problem at the tth iteration:

$$\begin{aligned} \min_{\boldsymbol{\alpha}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \epsilon, \boldsymbol{\delta}} \quad & \frac{1}{2}\boldsymbol{\alpha}'\mathbf{K}\boldsymbol{\alpha} + \gamma\nu\epsilon + \frac{\gamma}{m}(\boldsymbol{\xi}'\mathbf{1} + \boldsymbol{\xi}^{*\prime}\mathbf{1}) + \frac{\gamma\lambda}{m}\boldsymbol{\delta}'\mathbf{1} \\ \text{s.t.} \quad & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} + b - y_i \le \epsilon + \xi_i, \\ & y_i - \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - b \le \epsilon + \xi_i^*, \\ & \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} - \delta_i \le \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha}, \\ & \mathbf{k}'_{\mathcal{I}(B_i)}\boldsymbol{\alpha} - \sum_{j=1}^{n_i} \beta_{ij}^{(t)} \mathbf{k}'_{\mathcal{I}(\mathbf{x}_{ij})}\boldsymbol{\alpha} \le \delta_i, \\ & \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\delta} \ge \mathbf{0}, \ \epsilon \ge 0, \end{aligned}$$

which is again a QP problem.

On the other hand, Ray and Page (2001) used the Expectation Maximization (EM) algorithm to find the primary instance. Note that this can also be easily incorporated into our formulation, by simply replacing $\max_{j=1,\ldots,n_i} f(\mathbf{x}_{ij})$ in (17) with $f(\hat{\mathbf{x}})$, where $\hat{\mathbf{x}}$ is the primary instance predicted by EM. In this case, the computation is even simpler as the non-smooth max function is removed, and the optimization problem is immediately a QP problem without using CCCP.

5. Experiments

As mentioned in Section 1, Gärtner et al. (2002) proposed a number of MI kernels. Unlike the standard kernels that are defined on instances, these kernels are defined on bags. In the following, we will use a particular MI kernel called the *normalized set kernel*. Given a kernel k defined on instances, and any two bags B_1 and B_2, the normalized set kernel $\kappa(\cdot,\cdot)$ is defined as

$$\kappa(B_1, B_2) = \frac{k_{set}(B_1, B_2)}{\sqrt{k_{set}(B_1, B_1)}\sqrt{k_{set}(B_2, B_2)}}, \quad (18)$$

where $k_{set}(B_1, B_2) = \sum_{\mathbf{x} \in B_1, \mathbf{z} \in B_2} k(\mathbf{x}, \mathbf{z})$. In the experiments, we will use the Gaussian kernel as k. Note that in the degenerate case where both bags contain only one instance, $\kappa(\{\mathbf{x}\}, \{\mathbf{z}\})$ reduces to the usual normalized kernel $k(\mathbf{x}, \mathbf{z})/\sqrt{k(\mathbf{x}, \mathbf{x})k(\mathbf{z}, \mathbf{z})}$.

Recall that a number of loss functions can be used as $\ell(\cdot,\cdot)$ in our method. In the sequel, we will always use the L1 loss (Section 3.3.1) in all the experiments.

5.1. Classification: Corel Images

The first experiment is on image categorization using the data set[4] in (Chen & Wang, 2004), which in turn is extracted from the COREL images. There are 10 classes (beach, flowers, horses, etc.), with each class containing 100 images. Each image is regarded as a bag, and is partitioned into a number of non-overlapping segments using the k-means clustering algorithm. Each segment, represented by 9 features, is then regarded as an instance. On average, there are about 4.3 instances per bag.

We follow exactly the same setup as (Chen & Wang, 2004). The data is randomly divided into a training and test set, each containing 50 images of each category. Since this is a multi-class problem, the standard one-vs-rest approach is taken. The experiment is repeated 5 times, and the average accuracy reported. Model parameters are selected using a validation set.

Comparisons are made with 1) DD-SVM (Chen & Wang, 2004), which is a SVM that learns a set of instance prototypes by using the diverse density (DD) function (Maron & Lozano-Perez, 1998); 2) Hist-SVM (Chapelle et al., 1999), which is a histogram-based SVM for image classification; 3) MI-SVM (Andrews et al., 2003); 4) Standard SVM using the MI kernel in (18). Note that in the Hist-SVM and MI-SVM, dual variables are only associated with the instances; whereas in the DD-SVM and standard SVM using the MI kernel, dual variables are only associated with the bags. Our method, on the other hand, has dual variables associated with both the bags and instances.

Table 1 shows the results. Recall that the main difference between our method and MI-SVM is in the loss function ℓ, the superiority of our method over MI-SVM thus demonstrates the usefulness of introducing more

[4] http://www.cs.uno.edu/∼yixin/ddsvm.html.

flexible loss functions. Moreover, our method also outperforms the standard SVM with MI kernel. To ensure that the improvement is statistically significant, we performed 50 repetitions (instead of 5) and the difference is confirmed to be significant at the 0.01 level of significance by using the paired t-test. As discussed in Section 3.1, this shows that explicitly considering the loss between the predictions of each bag and its constituent instances is useful.

Table 1. Performance of MI classification on the Corel data set. Here, results on DD-SVM, Hist-SVM and MI-SVM are taken from (Chen & Wang, 2004).

	method	accuracy (%)
averaged over 5 repetitions	DD-SVM	81.5 ± 3.0
	Hist-SVM	66.7 ± 2.2
	MI-SVM	74.7 ± 0.6
	SVM (MI kernel)	84.1 ± 0.90
	our method	84.4 ± 1.38
averaged over 50 repetitions	SVM (MI kernel)	84.11 ± 1.23
	our method	85.17 ± 1.29

5.2. Regression: Synthetic Musk Molecules

This synthetic data set[5] is generated by Dooly et al. (2002) using an affinity model between the musk molecules and receptors. The goal is to predict the real-valued binding energies of these molecules. In this experiment, we use three data sets[6] (LJ-16.30.2, LJ-80.166.1 and LJ-160.166.1) downloaded from the author's website[7]. There are four more datasets available there. However, they are easier variations of the three that we used, and so are dropped in this study.

These three data sets are relatively low-dimensional. To make the regression problem more challenging, we created three more data sets (LJ-16-50-2, LJ-80-206-1 and LJ-160-566-1) by adding irrelevant features to the original data sets. For example, LJ-16-50-2 is generated by adding 20 more irrelevant features to LJ-16-30-2 while keeping its real-valued outputs intact.

Comparisons are made with the following MI regression methods: 1) Diverse Density (DD) (Maron & Lozano-Perez, 1998); 2) EM-DD (Zhang & Goldman, 2002); 3) Citation-kNN (Wang & Zucker, 2000); 4) Standard SVM using the MI kernel in (18). Recently, Ray and Craven (2005) demonstrated that supervised learners can sometimes outperform MI methods on MI (classification) problems. However, we speculate that this may be less likely for MI regression. Thus, we also compare with a standard SVM using an instance-level kernel, and the instances inherit their real-valued outputs from the corresponding bags. As in (Dooly et al., 2002), we will report the percentage error (%err)[8] and mean squared error (MSE).

Table 2 shows the results. On the first three data sets, the proposed method achieves comparable or better performance as the citation-kNN and MI kernel-based SVM. On the other hand, DD does not perform well, and so is dropped on the last three data sets that are more difficult. As pointed out in (Amar et al., 2001), the performance of the nearest neighbor-based and DD algorithms degrade rapidly as the fraction of relevant features decreases. This is demonstrated by the results on the last three data sets, where our SVM-based approach has consistently the best performance. Moreover, as expected, the supervised method does not perform well in MI regression.

6. Conclusion

In this paper, we provide a more complete regularization framework for MI learning by allowing a loss function between the outputs of a bag and its associated instances. This is less restrictive than strictly constraining the two to be equal as in previous studies, and is particularly useful when the bag labels are noisy or in MI regression. This new framework allows the use of different loss functions for MI classification and regression, and both bags and instances can now directly participate in the optimization process. Moreover, instead of using heuristics to solve the resultant nonlinear optimization problem, we use the constrained concave-convex procedure, and thus guarantees the convergence to local optimum. Experiments on both classification and regression data show that the proposed method can have improved performance.

Acknowledgments

This research has been partially supported by the Research Grants Council of the Hong Kong Special Administrative Region under grant 615005.

References

Amar, R., Dooly, D., Goldman, S., & Zhang, Q. (2001). Multiple-instance learning of real-valued

[5] As far as we know, "Affinity" is the only real-world data set used in MI regression experiments (Dooly et al., 2002). However, this data set is not publicly available.

[6] We follow the naming convention in (Dooly et al., 2002). Each filename is of the form LJ-r-f-s, where r is the number of relevant features, f is the total number of features, and s is the number of different scale factors used for the relevant features.

[7] http://www.cs.wustl.edu/∼sg/multi-inst-data

[8] %err is the classification error obtained by thresholding both the target output and the predicted output at 0.5.

Table 2. Performance of MI regression on the synthetic affinity data set. Here, results of DD and citation-kNN on the first three data sets are taken from (Dooly et al., 2002).

data set	DD		EM-DD		citation-kNN		SVM (MI kernel)		our method		supervised method	
	%err	MSE	%err	MSE	%err	MSE	%err	MSE	%err	MSE	%err	MSE
LJ-16.30.2	6.7	0.0240	6.7	**0.0153**	16.7	0.0260	10	0.0184	10	0.0185	45	0.0699
LJ-80.166.1	(not available)		37.0	0.1825	8.6	0.0109	8.7	0.0135	4.3	**0.0097**	33.7	0.0488
LJ-160.166.1	23.9	0.0852	21.7	0.0850	4.3	**0.0014**	0	0.0054	0	0.0053	2.2	0.0264
LJ-16-50-2	-	-	40	0.2357	53.3	0.0916	40	0.0723	33.3	**0.0673**	40	0.0716
LJ-80-206-1	-	-	37.0	0.2449	30.4	0.0463	23.9	0.0325	22.8	**0.0321**	18.5	0.0519
LJ-160-566-1	-	-	37.0	0.2414	34.8	0.0566	37.0	0.0535	33.7	**0.0507**	37.0	0.0570

data. *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 3–10). Williamstown, MA, USA.

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2003). Support vector machines for multiple-instance learning. *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Chapelle, O., Haffner, P., & Vapnik, V. (1999). Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, *10*, 1055–1064.

Chen, Y., & Wang, J. (2004). Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, *5*, 913–939.

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

Dooly, D., Zhang, Q., Goldman, S., & Amar, R. (2002). Multiple-instance learning of real-valued data. *Journal of Machine Learning Research*, *3*, 651–678.

Gärtner, T., Flach, P., Kowalczyk, A., & Smola, A. (2002). Multi-instance kernels. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 179–186). Sydney, Australia.

Maron, O., & Lozano-Perez, T. (1998). A framework for multiple-instance learning. In M. Jordan, M. Kearns and S. Solla (Eds.), *Advances in neural information processing systems 10*. San Mateo, CA: Morgan Kaufmann.

Maron, O., & Ratan, A. L. (1998). Multiple-instance learning for natural scene classification. *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 341–349). Madison, Wisconson, USA.

Murray, J. F., Hughes, G. F., & Kreutz-Delgado, K. (2005). Machine learning methods for predicting failures in hard drives: A multiple-instance application. *Journal of Machine Learning Research*, *6*, 783–816.

Pham Dinh, T., & Hoai An, L. (1998). A D.C. optimization algorithm for solving the trust-region subproblem. *SIAM Journal on Optimization*, *8*, 476–505.

Ray, S., & Craven, M. (2005). Supervised versus multiple instance learning: An empirical comparison. *Proceedings of the Twentieth-Second International Conference on Machine Learning* (pp. 697–704). Bonn, Germany.

Ray, S., & Page, D. (2001). Multiple instance regression. *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 425–432). Williamstown, MA, USA.

Schölkopf, B., & Smola, A. (2002). *Learning with kernels*. Cambridge, MA: MIT Press.

Smola, A. J., Vishwanathan, S. V. N., & Hofmann, T. (2005). Kernel methods for missing variables. *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*. Barbados.

Tao, Q., Scott, S., Vinodchandran, N., & Osugi, T. (2004). SVM-based generalized multiple-instance learning via approximate box counting. *Proceedings of the International Conference on Machine Learning*.

Wang, J., & Zucker, J.-D. (2000). Solving multiple-instance problem: A lazy learning approach. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 1119–1125). Stanford, CA, USA.

Yuille, A., & Rangarajan, A. (2003). The concave-convex procedure. *Neural Computation*, *15*, 915–936.

Zhang, Q., & Goldman, S. (2002). EM-DD: An improved multiple-instance learning. *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press.