Online Multiple Instance Learning with No Regret

Mu Li[1, 2] James T. Kwok[2] Bao-Liang Lu[1, 3]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[2]Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong
[3]MOE-Microsoft Key Laboratory for Intelligent Computing and Intelligent Systems
Shanghai Jiao Tong University, Shanghai 200240, China
limu@sjtu.edu.cn jamesk@cse.ust.hk bllu@sjtu.edu.cn

Abstract

Multiple instance (MI) learning is a recent learning paradigm that is more flexible than standard supervised learning algorithms in the handling of label ambiguity. It has been used in a wide range of applications including image classification, object detection and object tracking. Typically, MI algorithms are trained in a batch setting in which the whole training set has to be available before training starts. However, in applications such as tracking, the classifier needs to be trained continuously as new frames arrive. Motivated by the empirical success of a batch MI algorithm called MILES, we propose in this paper an online MI learning algorithm that has an efficient online update procedure and also performs joint feature selection and classification as MILES. Besides, while existing online MI algorithms lack theoretical properties, we prove that the proposed online algorithm has a (cumulative) regret of $\mathcal{O}(\sqrt{T})$, where T is the number of iterations. In other words, the average regret goes to zero asymptotically and it thus achieves the same performance as the best solution in hindsight. Experiments on a number of MI classification and object tracking data sets demonstrate encouraging results.

1. Introduction

In traditional supervised learning, each training pattern is associated with a known class label. However, in many real-world applications, the available label information is often weak and ambiguous. In this paper, we focus on a recent machine learning paradigm known as *multiple instance* (MI) learning. Here, concepts are learned from collections (called *bags*) of instances rather than from instances. Only the bags, but not the instances, have known labels. The so-called MI assumption then relates the bag labels to the unknown instance labels: A bag is labeled positive when at least one of its instances is positive; and a bag is negative when all its instances are negative.

The most famous MI application is drug activity prediction, which is introduced in the seminal work of Dietterich *et al*. [6]. Each drug molecule (considered as a bag) has multiple low-energy conformations (instances), and is considered useful as a drug if one of its conformations can bind to the targets. Another well-known MI application is content-based image classification and retrieval [7, 14]. Each image (bag) has a number of local patches (instances) and is considered relevant to the user query when at least one of these patches is relevant. Recently, Viola *et al*. [13] pioneered the use of MI learning in object detection, and obtained significantly improved detection rate. Here, a positive bag contains image patches that are near the labeled object. Very recently, Babenko *et al*. [3] further extended this for learning the appearance model in object tracking. Other MI applications include computer hard-drive failure prediction [8], protein classification [12], and text categorization [2].

Dietterich *et al*. used axis-parallel rectangles for MI learning in their seminal work. Following this, a wide range of approaches, including decision trees, Bayesian methods, ensemble methods and kernel methods, have emerged. In this paper, we will focus on a recent algorithm called MILES (Multiple-Instance Learning via Embedded Instance Selection) [5]. It maps each bag to a feature space defined by all the instances in the training bags, and then performs joint feature selection and classification by using the 1-norm SVM [15]. Empirically, it is highly efficient, accurate and robust to label noise.

Typically, MI algorithms are trained in a batch setting, in which the whole training set has to be available before training starts. However, in applications such as tracking, the classifier needs to be trained continuously as new frames arrive. Very recently, Babenko *et al*. [3] proposed an online MI algorithm based on boosting, and obtained encouraging object tracking results on several challenging video sequences. However, in training its weak classifiers during the boosting process, it imposes a strong assumption that all the instances in a positive bag are positive. This is often violated in many MI applications. Moreover, the weak classifiers are trained based only on the current frame and is thus susceptible to over-fitting.

Besides, for online algorithms in general, one of the most important quality metrics is the (cumulative) regret, which measures the gap between the cumulative loss of the online algorithm and that of the optimal solution in hindsight. In particular, if the regret is $o(T)$, where T is the number of iterations, the average regret becomes $o(1)$ and the online algorithm is thus guaranteed to be asymptotically optimal. However, though the MI learning algorithm proposed in [3] is an online algorithm, it does not have any known regret bound. Indeed, there has been no theoretical study even on assuring its convergence.

In this paper, we adopt the algorithmic framework for online learning in [11] and extend this for online MI learning. The proposed algorithm is motivated from MILES because of its empirical superiority. However, the ℓ_1-regularizer underlying the 1-norm SVM in MILES does not satisfy the strong convexity requirement in [11]. Thus, we propose instead the use of the elastic net regularizer [16]. We develop an efficient and general online MI learning algorithm with a regret bound of $\mathcal{O}(\sqrt{T})$. Besides, because of the feature map construction method in MILES, its feature vectors become variable length in an online setting. This also has to be adapted in order for the theoretical development to proceed.

The rest of this paper is organized as follows. Section 2 first gives brief introductions on MI learning and online learning. The proposed online MI learning algorithm is presented in Section 3. Finally, experimental results are presented in Section 4, and the last section gives some concluding remarks. All the proofs are in the appendix.

Notation For any vector $\mathbf{x}$, its transpose is denoted $\mathbf{x}'$, and its ith component by $\mathbf{x}[i]$. The 1-norm of $\mathbf{x}$ is $\|\mathbf{x}\|_1 = \sum_i |x[i]|$, and its 2-norm is $\|\mathbf{x}\|_2 = \sqrt{\sum_i (x[i])^2}$. The thresholding function is defined as $(z)_+ = \max(0, z)$. For notational simplicity, we use operations between vector and scalar to mean operating the scalar on each element of the vector. For example, "$\mathbf{w} + c$" means "$\mathbf{w}[i] + c$ for each component of $\mathbf{w}$". Similarly, $(\mathbf{w})_+$ means thresholding $\mathbf{w}$ component-by-component with the $(\cdot)_+$ function.

2. Previous Works

2.1. Multiple Instance Learning using MILES

In MI classification, we are given a set of training bags $\{(\mathcal{B}_1, y_1), \ldots, (\mathcal{B}_m, y_m)\}$, where $\mathcal{B}_i$ is the ith bag containing instances $\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in_i}$, and $y_i \in \pm 1$. Unlike supervised learning, only the bag labels, but not those of the individual instances, are available in MI learning.

MILES [5] converts a MI learning problem to a standard supervised learning problem as follows. First, a similarity measure $s(\cdot, \cdot)$ between two instances is defined. For example, $s(\mathbf{x}_k, \mathbf{x}_{k'}) = \exp\left(-\frac{1}{\sigma^2}\|\mathbf{x}_k - \mathbf{x}_{k'}\|_2^2\right)$ has been commonly used. Then, one can define the similarity between an instance $\mathbf{x}_k$ and a bag $\mathcal{B}_i$ as $s(\mathbf{x}_k, \mathcal{B}_i) = \max_{j=1,\ldots,n_i} s(\mathbf{x}_k, \mathbf{x}_{ij})$. By using the set of all instances in the training bags $\mathcal{C} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ (where $N = \sum_{i=1}^m n_i$), bag $\mathcal{B}_i$ is then mapped to the feature vector

$$\boldsymbol{z}_i = [s(\mathbf{x}_1, \mathcal{B}_i), s(\mathbf{x}_2, \mathcal{B}_i), \cdots, s(\mathbf{x}_N, \mathcal{B}_i)]'. \quad (1)$$

Intuitively, each instance in $\mathcal{C}$ represents a candidate concept in the MI problem, and the kth element of $\boldsymbol{z}_i$ represents the similarity between concept $\mathbf{x}_k$ and bag $\mathcal{B}_i$.

Since the feature vector $\boldsymbol{z}_i$ in (1) is potentially very long, MILES uses the 1-norm SVM to perform feature selection and classifier construction simultaneously. The 1-norm SVM can be formulated as the following optimization problem:

$$\min_{\mathbf{w}} \ \alpha\|\mathbf{w}\|_1 + \sum_{i=1}^m g_i(\mathbf{w}), \quad (2)$$

where

$$g_i(\mathbf{w}) = \begin{cases} C_1(1 - y_i\mathbf{w}'\boldsymbol{z}_i)_+ & y_i = 1 \\ C_2(1 - y_i\mathbf{w}'\boldsymbol{z}_i)_+ & y_i = -1 \end{cases}$$

is the hinge loss for the ith bag, and C_1, C_2 are regularization parameters used to penalize errors on the positive and negative bags, respectively. Moreover, it is well-known that the ℓ_1-regularizer in (2) encourages sparsity.

2.2. Online Learning

In online learning, we minimize the regularized loss over a period of T, i.e.,

$$\inf_{\mathbf{w}} \left(cf(\mathbf{w}) + \sum_{t=1}^T g_t(\mathbf{w}) \right), \quad (3)$$

where $f(\mathbf{w})$ is the regularizer (e.g., ℓ_2-regularizer), $g_t(\mathbf{w})$ is the loss function (e.g., hinge loss) at time t, and $c > 0$ is the regularization parameter. We assume that both f and g_t's are convex, and that $\inf_{\mathbf{w}} f(\mathbf{w}) = \inf_{\mathbf{w}} g_t(\mathbf{w}) = 0$. Usually, these are satisfied by typical regularizers and losses. It can be shown that the dual objective can be obtained as [11]

$$\mathcal{D}(\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_T) = -cf^\star\left(-\frac{1}{c}\sum_{t=1}^T \boldsymbol{\lambda}_t\right) - \sum_{t=1}^T g_t^\star(\boldsymbol{\lambda}_t), \quad (4)$$

where $\boldsymbol{\lambda}_t$ is the vector of Lagrangian multipliers at time t, and $f^\star$ ($g^\star$ resp.) is the Fenchel conjugate of f (g resp.):

$$f^\star(\boldsymbol{\lambda}) = \sup_{\mathbf{w}}(\mathbf{w}'\boldsymbol{\lambda} - f(\mathbf{w})).$$

To maximize the dual objective, Shalev-Shwartz and Singer [11] proposed the following online learning algorithm when f is strongly convex[1]. Let the estimate of $\boldsymbol{\lambda}_i$ at time t be $\boldsymbol{\lambda}_i^{(t)}$. Initially, all the $\{\boldsymbol{\lambda}_i^{(0)}\}_{i=1}^T$ are set to $\mathbf{0}$. Then, at time t, $\{\boldsymbol{\lambda}_i^{(t+1)}\}_{i=1}^T$ are updated such that

$$\begin{aligned}\exists \tilde{\boldsymbol{\lambda}} \in \partial g_t(\mathbf{w}_t), \text{s.t. } \mathcal{D}(\boldsymbol{\lambda}_1^{(t+1)}, \cdots, \boldsymbol{\lambda}_t^{(t+1)}, \mathbf{0}, \cdots, \mathbf{0}) \\ \geq \mathcal{D}(\boldsymbol{\lambda}_1^{(t)}, \cdots, \boldsymbol{\lambda}_{t-1}^{(t)}, \tilde{\boldsymbol{\lambda}}, \mathbf{0}, \cdots, \mathbf{0}).\end{aligned} \quad (5)$$

Moreover, the solution of (3) is obtained as

$$\mathbf{w}_t = \nabla f^\star \left(-\frac{1}{c} \sum_{i=1}^T \boldsymbol{\lambda}_i^{(t)} \right). \quad (6)$$

The quality of an online learning algorithm is usually measured by its *regret*, which is the loss for not consistently using a vector $\mathbf{w}$. As shown by the following theorem, this algorithm achieves $O(\sqrt{T})$ regret.

Theorem 1. *[11] Suppose that the regularizer f is μ-strongly convex. Let $L = \frac{1}{T}\sum_{t=1}^T \|\tilde{\boldsymbol{\lambda}}_t\|^2$, where $\tilde{\boldsymbol{\lambda}}_t \in \partial g_t(\mathbf{w}_t)$ for all t. Then, for any $\mathbf{w}$,*

$$\sum_{t=1}^T g_t(\mathbf{w}_t) - \sum_{t=1}^T g_t(\mathbf{w}) \leq cf(\mathbf{w}) + \frac{LT}{2\mu c}.$$

In particular, on setting $c = \sqrt{T}$,

$$\sum_{t=1}^T g_t(\mathbf{w}_t) - \sum_{t=1}^T g_t(\mathbf{w}) \leq (f(\mathbf{w}) + L/(2\mu))\sqrt{T}. \quad (7)$$

Consequently, the average regret for each step is $O(\sqrt{T}/T) = O(1/\sqrt{T})$, and thus goes to zero as $T \to \infty$.

3. Online MI Learning

3.1. Elastic Net Regularizer

Recall from Section 2.2 that the regularizer $f(\cdot)$ has to be strongly convex. However, for the 1-norm SVM used in MILES (Section 2.1), its ℓ_1-regularizer is not. In the following, we will use instead the elastic net regularizer [16], which is a combination of the ℓ_1- and ℓ_2-norms:

$$f(\mathbf{w}) = \alpha\|\mathbf{w}\|_1 + \frac{\beta}{2}\|\mathbf{w}\|_2^2, \quad (8)$$

with $\alpha \geq 0$ and $\beta > 0$. It is easy to see that f is β-strongly convex. Moreover, similar to the ℓ_1-regularizer, the elastic net regularizer can also lead to a sparse solution [16].

Recall that the online learning algorithm in [11] requires $f^\star(\boldsymbol{\lambda}), \nabla f^\star(\boldsymbol{\lambda})$ and $g^\star(\boldsymbol{\lambda})$. These can be readily computed by using the following two propositions.

[1]A function $f(\mathbf{w})$ is μ-strongly convex if $f(\mathbf{v}) \geq f(\mathbf{u}) + g(\mathbf{u})'(\mathbf{v} - \mathbf{u}) + \frac{\mu}{2}\|\mathbf{v} - \mathbf{u}\|_2^2$ for any $\mathbf{u}, \mathbf{v}$ and subgradient $g(\mathbf{u}) \in \partial f(\mathbf{u})$.

Proposition 1. *For the elastic net regularizer in (8),*

$$f^\star(\boldsymbol{\lambda}) = \frac{\|(|\boldsymbol{\lambda}| - \alpha)_+\|_2^2}{2\beta}, \ \nabla f^\star(\boldsymbol{\lambda}) = \frac{\text{sgn}(\boldsymbol{\lambda})(|\boldsymbol{\lambda}| - \alpha)_+}{\beta}.$$

Proposition 2. *For $g(\mathbf{w}) = C(1 - y\mathbf{w}'\boldsymbol{z})_+$, where $C > 0$,*

$$g^\star(\boldsymbol{\lambda}) = \begin{cases} \theta & \boldsymbol{\lambda} = \theta y \boldsymbol{z}, \text{ where } \theta \in [-C, 0], \\ \infty & \text{otherwise.} \end{cases} \quad (9)$$

Their proofs are in the appendix.

3.2. Efficient Update of $\boldsymbol{\lambda}_i^{(t+1)}$'s

We adopt the algorithmic framework of [11] in Section 2.2. Since the regularization parameter c in (3) has been absorbed into the elastic regularizer of (8), we can simply set $c = 1$. In general, there are various ways of finding $\{\boldsymbol{\lambda}_i^{(t+1)}\}_{i=1}^T$ that satisfy (5). For example, as suggested in [11], one can set

$$\boldsymbol{\lambda}_i^{(t+1)} = \begin{cases} \tilde{\boldsymbol{\lambda}} & i = t, \\ \boldsymbol{\lambda}_i^{(t)} & i \neq t, \end{cases} \quad (10)$$

where $\tilde{\boldsymbol{\lambda}} \in \partial g_t(\mathbf{w}_t)$. However, a faster increase in the dual objective value $D(\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_T)$ will lead to a more aggressive update and thus possibly faster convergence. Hence, we will obtain $\boldsymbol{\lambda}_t^{(t+1)} = \tilde{\boldsymbol{\lambda}}$ in (10) as

$$\begin{aligned}\boldsymbol{\lambda}_t^{(t+1)} &= \arg\max_{\boldsymbol{\lambda}} \mathcal{D}(\boldsymbol{\lambda}_1^{(t)}, \cdots, \boldsymbol{\lambda}_{t-1}^{(t)}, \boldsymbol{\lambda}, \mathbf{0}, \cdots, \mathbf{0}) \\ &= \arg\max_{\boldsymbol{\lambda}} -f^\star(\boldsymbol{\pi}_t + \boldsymbol{\lambda}) - g_t^\star(\boldsymbol{\lambda}),\end{aligned} \quad (11)$$

where

$$\boldsymbol{\pi}_t = \sum_{i<t} \boldsymbol{\lambda}_i^{(t)}. \quad (12)$$

Obviously, this also satisfies (5). By using Proposition 2, and define

$$\eta = \begin{cases} -C_1 & y_t = 1, \\ -C_2 & y_t = -1, \end{cases} \quad (13)$$

we obtain

$$\boldsymbol{\lambda}_t^{(t+1)} = \tilde{\theta} y_t \boldsymbol{z}_t, \quad (14)$$

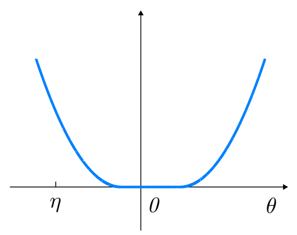
where

$$\begin{aligned}\tilde{\theta} &= \arg\max_{\theta\in[\eta,0]} -f^\star(\boldsymbol{\pi}_t + \theta y_t \boldsymbol{z}_t) - \theta \\ &= \arg\min_{\theta\in[\eta,0]} \frac{1}{2}\|(|\boldsymbol{\pi}_t + \theta y_t \boldsymbol{z}_t| - \alpha)_+\|_2^2 + \beta\theta,\end{aligned} \quad (15)$$

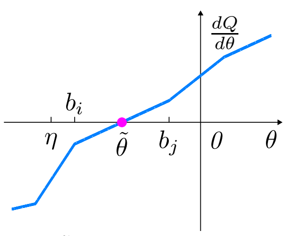
on using Proposition 1.

This minimization problem can be solved efficiently as follows. Denote the objective in (15) by $Q(\theta)$, which can be rewritten as

$$Q(\theta) = \frac{1}{2}\sum_i (|\boldsymbol{\pi}_t[i] + \theta y_t \boldsymbol{z}_t[i]| - \alpha)_+^2 + \beta\theta. \quad (16)$$



(a) A term inside the summation.



(b) $\tilde{\theta}$ is between b_i and b_j.

Figure 1. The objective $Q(\theta)$ in (16).

Note that each term in the summation is of the form shown in Figure 1(a) and has two breakpoints, $-\frac{\alpha}{|y_t \boldsymbol{z}_t[i]|} - \frac{\boldsymbol{\pi}_t[i]}{y_t \boldsymbol{z}_t[i]}$ and $\frac{\alpha}{|y_t \boldsymbol{z}_t[i]|} - \frac{\boldsymbol{\pi}_t[i]}{y_t \boldsymbol{z}_t[i]}$.

Lemma 1. *$\frac{dQ}{d\theta}$ is piecewise linear and non-decreasing in θ.*

By using this lemma, (15) can be reduced to minimizing $\left|\frac{dQ}{d\theta}\right|$ over $\theta \in [\eta, 0]$. If the curve of $\left|\frac{dQ}{d\theta}\right|$ is entirely above/below zero for $\theta \in [\eta, 0]$, then $\tilde{\theta}$ is simply equal to either η or 0, depending on which one has the smaller $\left|\frac{dQ}{d\theta}\right|$ value. Otherwise, $\tilde{\theta}$ must lie between two breakpoints b_i and b_j, such that $\eta \le b_i < b_j \le 0$ and $\frac{dQ}{d\theta}\big|_{b_i} \frac{dQ}{d\theta}\big|_{b_j} \le 0$ (Figure 1(b)). By interpolation, we have $\tilde{\theta} = \left(b_i \frac{dQ}{d\theta}\big|_{b_i} + b_j \frac{dQ}{d\theta}\big|_{b_j}\right) \Big/ \left(\frac{dQ}{d\theta}\big|_{b_i} + \frac{dQ}{d\theta}\big|_{b_j}\right)$.

3.3. Variable-Length Feature Vectors

Recall that the feature map of MILES in (1) is defined in terms of the similarities with all the instances in all the training bags. In a batch setting, this whole set of instances is known and fixed before training starts. However, in an online setting, both the set of bags and the set of instances increase with time. Hence, the feature vectors at different t's are of different sizes. Specifically, $\boldsymbol{z}_t \in \mathbb{R}^{N_t}$, where $N_t = \sum_{i=1}^t n_i$. This poses a problem on attempting to use the algorithmic framework of [11].

To alleviate this problem, we extend all the feature vectors to some fictitious length N by appending zeros at the end. Note that this is only needed for the theoretical development, and N does not need to be known in practice. The following proposition shows that at time t, the last $(N - N_t)$ elements of $\boldsymbol{\lambda}_t$ and $\mathbf{w}_t$ are also 0.

Proposition 3. *The last $(N - N_t)$ elements of $\boldsymbol{\lambda}_t^{(t+1)}$ and $\mathbf{w}_t$ are zero.*

Hence, this is consistent with the construction that these entries are not needed at time t. With this appending of zeros, we can regard the feature vectors to be of fixed length.

3.4. The Complete Algorithm

The proposed online learning algorithm, which will be called MIO ("Multiple Instance Online"), is shown in Algorithm 1. Note that in step 3, $\mathbf{w}_t$ is computed from $\nabla f^\star(\cdot)$ in Proposition 1. This involves a soft-thresholding operation $(\cdot - \alpha)_+$, and hence the $\mathbf{w}_t$ obtained is sparse.

Algorithm 1 MIO ("Multiple Instance Online").

```
1: $\boldsymbol{\pi}_1 = \mathbf{0} \in \mathbb{R}^{n_1}$.
2: for $t = 1$ to $T$ do
3:     Compute $\mathbf{w}_t \leftarrow \nabla f^\star(\boldsymbol{\pi}_t)$ to predict on input $\boldsymbol{z}_t$.
4:     Receive the label $y_t$.
5:     $\theta_t \leftarrow \arg\min_{\theta \in [\eta, 0]} \|(|\boldsymbol{\pi}_t + \theta y_t \boldsymbol{z}_t| - \alpha)_+\|_2^2 + 2\beta\theta$,
       where $\eta$ is as defined in (13).
6:     $\boldsymbol{\lambda}_t \leftarrow \theta_t y_t \boldsymbol{z}_t$.
7:     $\boldsymbol{\pi}_{t+1} \leftarrow \begin{bmatrix} \boldsymbol{\pi}_t + \boldsymbol{\lambda}_t \\ \mathbf{0}_{n_{t+1}} \end{bmatrix}$.
8: end for
```

3.4.1 Regret Guarantee

Recall from Section 3.1 that the elastic net regularizer $f(\cdot)$ is β-strongly convex. By applying Theorem 1, we have $\sum_{t=1}^T g_t(\mathbf{w}_t) - \sum_{t=1}^T g_t(\mathbf{w}) \le f(\mathbf{w}) + \frac{LT}{2\beta}$ for any $\mathbf{w}$. In particular, on setting $\alpha = \alpha_0\sqrt{T}$ and $\beta = \beta_0\sqrt{T}$ for some $\alpha_0, \beta_0 \in \mathbb{R}$,

$$\sum_{t=1}^T g_t(\mathbf{w}_t) - \sum_{t=1}^T g_t(\mathbf{w}) \le \left(f_0(\mathbf{w}) + \frac{L}{2\beta_0}\right)\sqrt{T},$$

where $f_0(\mathbf{w}) = \alpha_0\|\mathbf{w}\|_1 + \frac{\beta_0}{2}\|\mathbf{w}\|_2^2$. In other words, $O(\sqrt{T})$ regret is also obtained. Asymptotically, the average regret w.r.t. the best $\mathbf{w}$ in hindsight goes to zero.

3.4.2 Time Complexity

Observe that $\mathbf{w}_t, \boldsymbol{\lambda}_t$ and $\boldsymbol{\pi}_t$ are vectors in $\mathbb{R}^{N_t}$. At time t, steps 3, 6, and 7 all take $\mathcal{O}(N_t)$ time. As for the minimization problem in step 5, the procedure in Section 3.2 involves sorting all the $2N_t$ breakpoints ($\mathcal{O}(N_t \log(N_t))$ time), and computing the gradients on all the breakpoints ($\mathcal{O}(N_t)$ time by a proper arrangement of the computations). Therefore, each update at time t takes a total of $\mathcal{O}(N_t \log(N_t))$ time.

4. Experiments

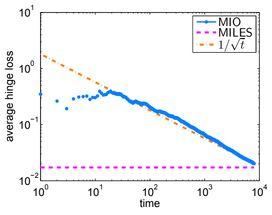
4.1. Synthetic Data

We first experiment with a two-dimensional synthetic data set used in [5]. Each instance is generated from one of the five normal distributions: $\mathcal{N}([5,5]', \mathbf{I})$, $\mathcal{N}([5,-5]', \mathbf{I})$, $\mathcal{N}([0,0]', \mathbf{I})$, $\mathcal{N}([-5,5]', \mathbf{I})$, and $\mathcal{N}([-5,-5]', \mathbf{I})$, where $\mathbf{I}$ is the identity matrix. Each bag has at most 8 instances. A bag is labeled positive if it contains instances from at least two of the first three distributions.

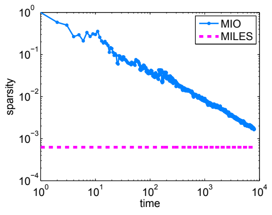
We compare MIO with MILES[2] on a data set with 4,000 positive bags and 4,000 negative bags. Figure 2(a) shows

[2]The code is downloaded from http://cs.olemiss.edu/~ychen/MILES.html

the resultant hinge loss values in log-log scale. Since the optimal weight cannot be explicitly determined, we regard the MILES solution as optimal. As can be seen, for sufficiently large T, the average regret of MIO decreases towards 0 at a rate of $1/\sqrt{T}$, which thus agrees with the regret bound in Section 3.4.1. Figure 2(b) shows the proportion of nonzero features in $\boldsymbol{z}_t$ that are selected by the classifier. As can be seen, the resultant feature representation is sparse and so MIO can also perform feature selection as MILES. However, it is not as sparse as MILES, which is also consistent with the observation that the elastic net tends to select more features than the ℓ_1-regularizer [16].



(a) Hinge loss values.



(b) Sparsity of features.

Figure 2. Performance of MIO on the synthetic data set. Note that both the abscissa and ordinate are in log-scale.

4.2. Drug Activity Prediction

In this section, experiments are performed on the popular Musk1 and Musk2 data sets from the UCI repository. The task is to predict if a drug (bag) is musky. The Musk1 data set contains 47 positive and 45 negative bags, with an average of 5.17 instances per bag; while the more challenging Musk2 data set contains 39 positive and 63 negative bags, with an average of 64.69 instances per bag. The experimental setup follows that in [5]. Fifteen runs of 10-fold cross-validation are performed and then the averaged performance reported. Since both musk data sets are small, MIO is cycled through the training data multiple times in each run to ensure sufficient convergence.

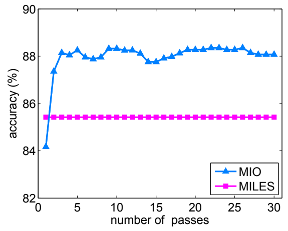
Figures 3(a) and 3(b) show the testing accuracies with different numbers of passes over the training data. As can be seen, MIO only requires a small number of passes to outperform MILES[3]. Table 1 also shows that it is competitive with other MI learning methods. Figures 3(c) and 3(d) show that the runtime of MIO increases linearly with the number of passes as expected. Note in particular that MIO is much faster on the larger Musk2 data. This is in line with the observation that online algorithms which sweep repetitively over the entire training set can be more computationally efficient than traditional batch learning algorithms [4].

4.3. Tracking

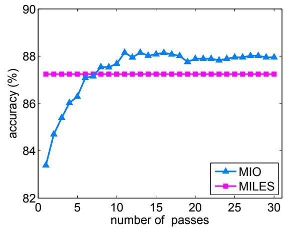
In this section, we follow [3] and use MI learning to adapt the appearance model for object tracking. Each positive bag contains a set of image patches that are close to the predicted object location; while each negative bag contains a patch that is unlikely to contain the object. As in [3], we represent each patch as a vector of Haar-like features [13]. Let the feature vector for patch p be $\mathbf{v}(p) = [v_i(p)]$. The similarity between patches p and p' is defined as $\exp\left(-\sum(v_i(p)-v_i(p'))^2/\sigma^2\right)$, where the summation is over the subset of i's with the k smallest $|v_i(p)-v_i(p')|$ values. Intuitively, this means two patches are similar if they have k similar Haar features [1]. In the experiments, k is set to 30.

Experiments are performed on the eight video sequences used in [3]. MIO is compared with MILTrack[4] [3] and FragTrack[5] [1], which is a very efficient algorithm using a static appearance model. Due to randomness in selecting

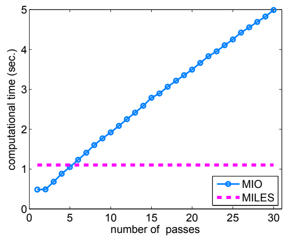
[3]The improvement is statistically significant (at the 5% level using the paired t-test) on Musk1, but not on Musk2.



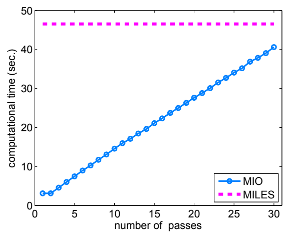
(a) Musk1.



(b) Musk2.



(c) Musk1.



(d) Musk2.

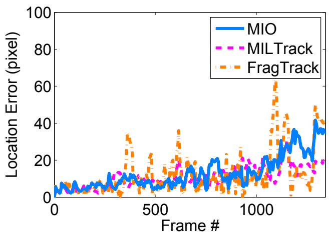
Figure 3. Performance of MIO on the musk data sets.

	Musk1	Musk2
MIO	88.3 : [86.5, 90.2]	87.7 : [85.8, 89.6]
MILES	85.5 : [83.0, 88.0]	87.1 : [82.3, 91.9]
APR	92.4	89.2
Bagging-APR	92.8	93.1
DD	88.9	82.5
DD-SVM	85.8	91.3
EM-DD	84.8	84.9
mi-SVM	87.4	83.6
MI-SVM	77.9	84.3
Multinst	76.7 : [73.6, 79.8]	84.0 : [81.4, 86.6]
RELIC	83.7	87.3

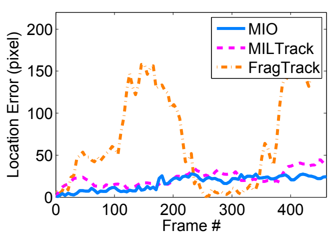
Table 1. Accuracies (in %) of the various MI learning algorithms on Musk1 and Musk2. Following [5], we report both the mean and 95% confidence interval for MIO (with 10 passes over training data) and MILES. Results for the other algorithms are from [5].

[4]Both the MILTrack code and videos are from `http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml`.

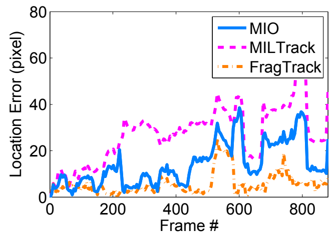
[5]`http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm`



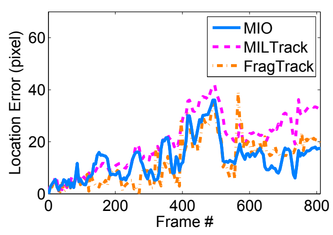
(a) Sylvester.



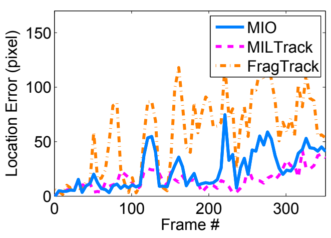
(b) David Indoor.



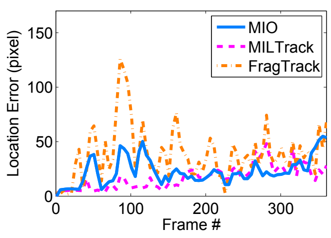
(c) Occluded Face.



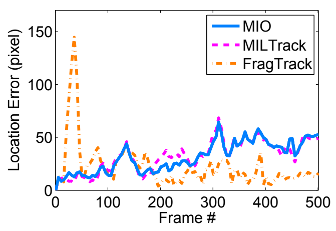
(d) Occluded Face 2.



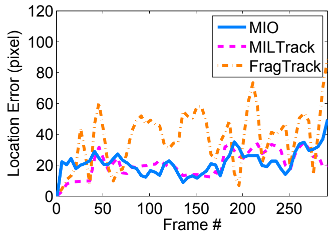
(e) Tiger 1.



(f) Tiger 2.

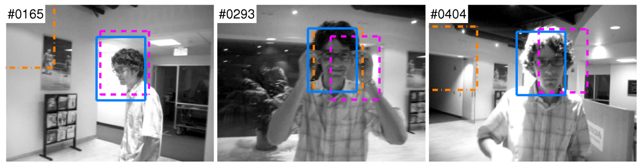


(g) Girl.

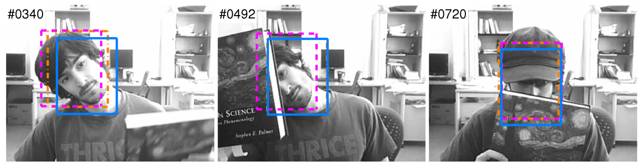


(h) Coke Can.

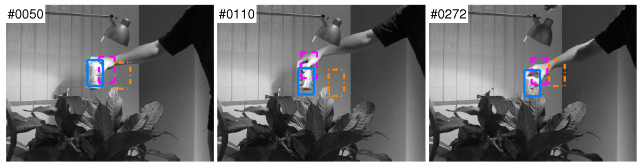
Figure 4. Errors plots for the video sequences.



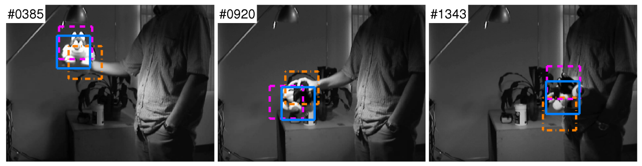
(a) David Indoor.



(b) Occluded Face 2.



(c) Coke Can.



(d) Sylvester.

Figure 5. Screenshots of the tracking results (MIO —; MILTrack - - -; FragTrack -.-.). For "David Indoor" and "Occluded Face 2", the same frames as in [3] are used.

the Haar-like features, each video clip is run five times as in [3] and then the averaged result reported.

Results are shown in Table 2 and Figure 4. Screenshots for some of the clips are also shown in Figure 5. As can be seen, MIO is competitive with and often better than MILTrack. FragTrack also performs particularly well on "Occlude Face" and "Occlude Face 2", as it is specifically designed to handle occlusions. However, sometimes it may completely lose track of the object (e.g., when the object performs a 180° rotation during frames 20-50 of the "Girl" sequence).

video clip	FragTrack	MILTrack	MIO
David Indoor	46	23	15
Sylvester	13	11	13
Occluded Face	6	27	14
Occluded Face 2	13	20	13
Girl	22	32	31
Tiger 1	56	15	24
Tiger 2	39	17	23
Coke Can	38	21	22

Table 2. Average center location errors (pixels).

5. Conclusions & Future Work

In this paper, we proposed an online MI learning algorithm that is motivated from the MILES algorithm. It uses the strongly convex elastic net regularizer, instead of the ℓ_1-regularizer, in the underlying classifier. We developed an efficient online update procedure and showed that it performs joint feature selection and classification as in MILES. Besides, we proved that the resultant online algorithm has a (cumulative) regret of $\mathcal{O}(\sqrt{T})$, and thus achieves the same performance as the best solution in hindsight.

While the algorithmic framework in [11] leads to a regret of $\mathcal{O}(\sqrt{T})$, recent research shows that it can be further accelerated to $\mathcal{O}(\log(T)/T)$ [10]. Preliminary results show that this can also be used to improve the proposed MIO algorithm, and will be further explored in the future.

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China (Grant No. 60773090 and Grant No. 90820018), the National Basic Research Program of China (Grant No. 2009CB320901), the National High-Tech Research Program of China (Grant No. 2008AA02Z315), and the Research Grants Council of the Hong Kong Special Administrative Region (Grant 615209).

A. Appendix

A.1. Proof of Proposition 1

Proof. First, consider the case where $\mathbf{w}$ is a scalar. (8) then reduces to $f(w) = \alpha|w| + \frac{\beta}{2}w^2$, and

$$\begin{aligned} f^\star(\lambda) &= \max_w \left(\lambda w - \alpha|w| - \frac{\beta}{2}w^2\right) \\ &= \max_w \left(\operatorname{sgn}(w)\lambda|w| - \alpha|w| - \frac{\beta}{2}|w|^2\right) \\ &= \max_{|w|} \left(|\lambda||w| - \alpha|w| - \frac{\beta}{2}|w|^2\right) \\ &= \max_{|w|} -\frac{(\beta|w| + \alpha - |\lambda|)^2}{2\beta} + \frac{(|\lambda| - \alpha)^2}{2\beta} \\ &= \begin{cases} 0 & \text{if } \alpha - |\lambda| \geq 0 \\ \frac{(|\lambda|-\alpha)^2}{2\beta} & \text{if } \alpha - |\lambda| < 0 \end{cases} = \frac{(|\lambda| - \alpha)_+^2}{2\beta} \end{aligned} \quad (17)$$

For the general case where $\mathbf{w} \in \mathbb{R}^n$,

$$\begin{aligned} f^\star(\boldsymbol{\lambda}) &= \max_{\mathbf{w}} \left(\boldsymbol{\lambda}'\mathbf{w} - \alpha\|\mathbf{w}\|_1 - \frac{\beta}{2}\|\mathbf{w}\|_2^2\right) \\ &= \sum_i \max_{\mathbf{w}[i]} \left(\boldsymbol{\lambda}[i]\mathbf{w}[i] - \alpha|\mathbf{w}[i]| - \frac{\beta}{2}\mathbf{w}[i]^2\right) \\ &= \sum_i \frac{(|\boldsymbol{\lambda}[i]| - \alpha)_+^2}{2\beta} = \frac{\|(|\boldsymbol{\lambda}| - \alpha)_+\|_2^2}{2\beta}, \end{aligned}$$

on using (17). Finally, $\nabla f^\star(\boldsymbol{\lambda})$ can be obtained from $f^\star(\boldsymbol{\lambda})$ by straightforward differentiation. □

A.2. Proof of Proposition 2

Proof. For the hinge loss $h(\mathbf{w}) = (1 - y\mathbf{w}'\boldsymbol{z})_+$, $h^\star(\boldsymbol{\lambda}) = \theta$ if $\boldsymbol{\lambda} = \theta y \boldsymbol{z}$ and $\theta \in [-1, 0]$; and $h^\star(\boldsymbol{\lambda}) = \infty$ otherwise [9]. Using the property that the Fenchel conjugate of $ah(\mathbf{w})$ is $ah^\star(\boldsymbol{\lambda}/a)$ [9], we can then obtain (9). □

A.3. Proof of Proposition 3

Proof. Since the last $(N - N_t)$ elements of $\boldsymbol{z}_t$ are zero, it is clear from (14) that the last $(N - N_t)$ elements of $\boldsymbol{\lambda}_t^{(t+1)}$ are also zero. Consequently, from (12), the last $(N - N_t)$ elements of $\boldsymbol{\pi}_t$ are also zero. In the following, we use the superscript o to denote the part excluding the last $(N - N_t)$ elements. From (6) and Proposition 1, we have

$$\begin{aligned} \mathbf{w}_t &= \nabla f^\star\left(-\sum_{i=1}^t \boldsymbol{\lambda}_i\right) = \nabla f^\star(-\boldsymbol{\pi}_t - \boldsymbol{\lambda}_t) \\ &= \nabla f^\star([-\boldsymbol{\pi}_t^o - \boldsymbol{\lambda}_t^o, \mathbf{0}_{(N-N_t)}]) \\ &= \begin{bmatrix} \nabla f^\star(-\boldsymbol{\pi}_t^o - \boldsymbol{\lambda}_t^o) \\ \mathbf{0}_{(N-N_t)} \end{bmatrix}. \end{aligned}$$

□

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, pages 798–805, 2006.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS 15*, pages 577–584, 2003.

[3] B. Babenko, M. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, pages 983–990, 2009.

[4] L. Bottou and Y. Le Cun. On-line learning for very large data sets. *Applied Stochastic Models in Business and Industry*, 21(2):137–151, March 2005.

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

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

[7] Z. Fu and A. Robles-Kelly. An instance selection approach to multiple instance learning. In *CVPR*, pages 911–918, 2009.

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

[9] R. Rifkin and R. Lippert. Value regularization and Fenchel duality. *Journal of Machine Learning Research*, 8:479, 2007.

[10] S. Shalev-Shwartz and S. Kakade. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS 21*, pages 1457–1464, 2009.

[11] S. Shalev-Shwartz and Y. Singer. Convex repeated games and Fenchel duality. In *NIPS 19*, pages 1265–1272, 2007.

[12] Q. Tao, S. Scott, N. Vinodchandran, and T. Osugi. SVM-based generalized multiple-instance learning via approximate box counting. In *ICML*, pages 101–108, 2004.

[13] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 511–518, 2001.

[14] Q. Zhang and S. Goldman. EM-DD: An improved multiple-instance learning technique. In *NIPS 14*, pages 1073–1080. MIT, 2002.

[15] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *NIPS 16*, pages 49–56, 2004.

[16] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.