Building Sparse Support Vector Machines for Multi-Instance Classification

Zhouyu Fu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang

Monash University, Churchill VIC 3842, Australia
{zhouyu.fu,guojun.lu,kaiming.ting,dengsheng.zhang}@monash.edu

Abstract. We propose a direct approach to learning sparse Support Vector Machine (SVM) prediction models for Multi-Instance (MI) classification. The proposed sparse SVM is based on a "label-mean" formulation of MI classification which takes the average of predictions of individual instances for bag-level prediction. This leads to a convex optimization problem, which is essential for the tractability of the optimization problem arising from the sparse SVM formulation we derived subsequently, as well as the validity of the optimization strategy we employed to solve it. Based on the "label-mean" formulation, we can build sparse SVM models for MI classification and explicitly control their sparsities by enforcing the maximum number of expansions allowed in the prediction function. An effective optimization strategy is adopted to solve the formulated sparse learning problem which involves the learning of both the classifier and the expansion vectors. Experimental results on benchmark data sets have demonstrated that the proposed approach is effective in building very sparse SVM models while achieving comparable performance to the state-of-the-art MI classifiers.

1 Introduction

Multi-instance (MI) classification is a paradigm in supervised learning first introduced by Dietterich [1]. In a MI classification problem, training examples are presented in the form of bags associated with binary labels. Each bag contains a collection of instances, whose labels are not perceived in priori. The key assumption for MI classification is that a positive bag contains at least one positive instance and a negative bag contains only negative instances.

We focus on SVM-based methods for MI classification. Existing methods include the MI-kernel [2], MI-SVM [3], mi-SVM [3], a regularization approach to MI [4], MILES [5], MissSVM [6], and KI-SVM [7]. Most existing approaches are based on the modification of standard SVM models and kernels in the single instance (SI) case to adapt to the MI setting. The SVM classifiers obtained from these methods take a similar analytic form as standard kernel SVMs. The prediction function can be expressed by a generalized linear model with predictor variables given by the kernel values evaluated for the Support Vectors (SVs), training instances with nonzero coefficients in the expansion. The speed of SVM prediction is proportional to the number of SVs. The sparser the SVM model

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part I, LNAI 6911, pp. 471–486, 2011.

(i.e. the smaller the number of SVs), the more efficient the prediction. The issue is more serious for MI classification for two reasons. Firstly, the number of SVs in the trained classifier largely depends on the number of training instances. Even with a moderate-size MI problem, a large number of instances may be encountered, consequently leading to a non-sparse SVM classifier with many SVs. Secondly, in MI classification, the prediction function is usually defined at instance-level either explicitly [3] or implicitly [2]. In order to predict the bag label, one needs to apply the classifier to each instance in the bag. Reducing a single SV in the prediction function would result in savings proportional to the size of the bag. Thus it is more important in MI prediction to remove redundant SVs for efficiency in prediction. For practical MI applications where fast prediction speed is required, it is highly desirable to have sparse SVM prediction models. Existing SVM solutions to MI classification are unlikely to produce a sparse prediction model in this respect. MILES [5] is the only exception, which produces a sparser solution than other methods, by using an L_1 norm on the coefficients to encourage the sparsity of the solution. However, it can not explicitly control the sparsity of the learned classifier.

In this paper, we propose a principled approach to learning sparse SVM for MI classification. The proposed approach can achieve controlled sparsity while maintaining competitive predictive performance as compared to existing non-sparse SVM models for MI classification. To the best of our knowledge, this is the first explicit formulation for sparse SVM learning in the MI setting. In the context of SI classification, sparse SVMs have attracted much attention in the research community [8–12]. The major difficulty to extend sparse learning to the MI scenario is the complexity of MI classification problems compared to SI ones. The MI assumption has to be explicitly taken into account in SVM modeling. This eventually leads to complicated SVM formulations with extra non-convex constraints and hence difficult optimization problems. To consider sparse MI learning, more constraints need to be considered. This further complicates the issue of modeling and leads to intractable optimization problems. Alternatively, one can use some pre- [9] or post-processing [10] schemes to empirically learn a sparse MI SVM. These schemes do not consider the special nature of an MI classification problem and are likely to yield inferior performance for prediction.

The method most closely related to ours was proposed in Wu et al. [12] for the SI case. Nevertheless, their method can not be directly applied to the MI case since the base model they considered is the standard SVM classifier. It is well-known that learning a standard SVM is equivalent to solving a convex quadratic problem [10] with the guarantee of unique global minimum. This is the pre-requisite for the optimization strategy employed in [12]. In contrast, existing SVM solutions to MI classification are either non-convex [3, 4, 6] or based on complex convex approximations [7]. It is non-trivial to extend the sparse learning framework [12] to MI classification based on the existing formulations. To this end, we adopt a simple "label-mean" formulation which takes the average value of instance label predictions for bag-level prediction. This is contrary to most existing formulations which makes bag-level prediction by taking the maximum

of instance predictions as indicated by the MI assumption. However, we justify the validity of the use of "label-mean" by showing theoretically that target MI concepts can be correctly classified via the proper choice of instance-level kernel function. This is established by linking it with the MI-kernel [2] and extending the theoretical results in [2]. Departing from the "label-mean" formulation, the sparse SVM for MI learning can then be formulated by imposing additional constraint on the classifier weight vector to explicitly control the complexity of the resulting prediction function. The proposed approach can simultaneously learn the MI SVM classifier as well as select the expansion vectors in the classifier function in a single discriminative framework. A similar optimization scheme to [12] can then be employed to effectively solve the newly formulated problem.

2 "Label-Mean" Formulation for MI Classification

In MI classification, we are given m training bags $\{B_1, \ldots, B_m\}$ and their corresponding labels $\{y_1, \ldots, y_m\}$ with $y_i \in \{-1, 1\}$, where $B_i = \{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,n_i}\}$ is the ith bag containing n_i instances. Each instance $\mathbf{x}_{i,p}$ in the bag is also associated with a hidden label $y_{i,p}$. By the MI assumption, $y_{i,p} = -1$ for all $p \in \{1, \ldots, n_i\}$ if $y_i = -1$ and $y_{i,p} = 1$ for some p if $y_i = 1$. Note that only bag labels are available for training and instance labels are not observable from the data. The relation between them can be captured by a single equation as follows

$$y_i = \max_{p=1}^{n_i} y_{i,p} \qquad \forall i \tag{1}$$

This naturally inspires a bottom-up "label-max" formulation for MI classification, which forms the backbone for many existing SVM-based formulations [3, 4, 6]. In this formulation, one aims at learning a prediction model at instance level and making the bag prediction by taking the maximum value of predictions obtained from individual instances. Specifically, the following condition is encoded for bag B_i in the SVM objective function either as hard [3, 6] or soft constraints [4] in accordance with the MI assumption in Equation 1

$$F(B_i) = \max_{p=1}^{n_i} f(\mathbf{x}_{i,p}) \tag{2}$$

where f and F denotes instance and bag-level prediction functions respectively. As a result, the above constraints lead to a non-convex and non-differentiable SVM model for MI classification and consequently result in a hard optimization problem. One has to resort to heuristics [3] or complicated optimization routines like the Concave-Convex Procedure (CCCP) [4, 6] to solve the hard optimization problems arising from these SVM models.

Here, we adopt an alternative SVM formulation for the MI classification problem. Instead of taking the maximum of instance predictions for bag-level prediction, we advocate the use of average operation over instance predictions. This leads to the following constraint for bag-level prediction

$$F(B_i) = \frac{1}{n_i} \sum_{p=1}^{n_i} f(\mathbf{x}_{i,p}) \tag{3}$$

Unlike the "label-max" constraints in Equation 2, the above constraints lead to a convex formulation for learning MI classifiers, which we term the "label-mean" formulation in the following.

We now formally define the optimization problem for the "label-mean" formulation. Let Ω denote the space of instance features, and $\mathcal{H}$ denote the Reproducing Kernel Hilbert Spaces (RKHS) of functions $f : \Omega \to \mathbb{R}$ induced by the kernel $k : \Omega \times \Omega \to \mathbb{R}$. As with the case of standard SVM, we consider instance-level linear prediction functions in the RKHS. Hence, we have

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{4}$$

where $\phi(\mathbf{x}) \in \mathcal{H}$ is the projection of instance $\mathbf{x}$ from input space Ω into the RKHS, $\mathbf{w} \in \mathcal{H}$ is the weight vector for the linear model in RKHS, and b is the bias term. Note that both $\mathbf{w}$ and $\phi(\mathbf{x})$ can be infinite dimensional depending on the mapping induced by the corresponding kernel k. The following optimization problem in RKHS is solved for MI classification

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{m} \ell(y_i, F(B_i)) \tag{5}$$

The first and second terms in the above objective function correspond to the regularization and data terms respectively, with parameter C controlling the trade-off between them. $\ell(y_i, F(B_i))$ is the bag-level loss function penalizing the discrepancy between bag label y_i and prediction $F(B_i)$ as given by Equation 3. We stick to 2-norm SVM throughout the paper, which uses the following squared Hinge loss function

$$\ell(y_i, F(B_i)) = \max\left(0, 1 - y_i F(B_i)\right)^2 \tag{6}$$

The purpose of the formulation in Equation 5 is to learn an instance-level prediction function f which gives good performance on bag-level prediction based on the "label-mean" relation in Equation 3. The objective function is continuously differentiable and convex with unique global minimum. Both properties are required for the successful application of the optimization strategy we used for solving the sparse SVMs proposed in the next section based on the "label-mean" formulation. The standard 1-norm SVM with the non-differentiable Hinge loss can also be used here, which was adopted in Wu et al. [12] for solving sparse SVM for SI learning. However, in this case, the optimization problem has to be solved in the dual formulation. This is computationally much more expensive than the primal formulation as we will discuss later.

The main difference between the proposed formulation and existing ones [3, 4, 6] is in the relation between $F(B_i)$, bag-level prediction, and $f(\mathbf{x}_{i,p})$'s, the prediction values for instances. We have made a small but important modification here by substituting the constraints in Equation 2 with those in Equation 3. This may look like a violation of the MI assumption in the beginning. Nevertheless, the rationale for taking the average will become self-evident once we reveal the relationship between the proposed formulation and the existing MI kernel method [2]. Specifically, we have the following lemma.

Lemma 1. *Training the SVM in Equation 5 is equivalent to training a standard 2-norm SVM at bag level with the normalized set kernel* $k_{set}(B_i, B_j) = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} k(\mathbf{x}_{i,p}, \mathbf{x}_{j,q})$*, where* k *is the instance-level kernel.*

Proof. By substituting Equations 3 and 4 into Equation 5 and introducing slack variables ξ_i's for the square Hinge losses, we can convert the unconstrained problem into a constrained quadratic program in the following

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i^2 \\ \text{s.t.} \quad & y_i\left(\frac{1}{n_i}\sum_{p=1}^{n_i} \mathbf{w}^T\phi(\mathbf{x}_{i,p}) + b\right) \geq 1 - \xi_i \qquad \forall i \end{aligned} \tag{7}$$

By making use of the Lagrangian and KKT conditions, we can derive the dual of the above problem in the following[1]

$$\begin{aligned} \max_{\alpha \geq 0} \quad & \sum_{i,j=1}^{m} \alpha_i y_i \alpha_j y_j K'_{i,j} - \sum_{i=1}^{m} \alpha_i \\ \text{s.t.} \quad & \sum_i y_i \alpha_i = 0 \end{aligned} \tag{8}$$

with $K'_{i,j} = k_{set}(B_i, B_j) + \frac{1}{2C}\delta_{i,j}$. This is exactly a 2-norm SVM for bags with kernel specified by k_{set}. □

With the above lemma, we can proceed to show the following theorem, which justifies the use of the mean operation for bag-level prediction in Equation 3.

Theorem 1. *If positive and negative instances are separable with margin* ϵ *with respect to feature map* ϕ *in the RKHS induced by kernel* k*, then for sufficiently large integer* r*, positive and negative bags are separable with margin* ϵ *using the bag-level prediction function in Equation 3 with instance kernel* k^r*.*

Proof. The proof follows directly from Lemma 4.2 in [2], which states that if positive and negative instances are separable with margin ϵ with respect to kernel k, then positive and negative bags can be separated with the same margin by the following MI kernel

$$k_{MI}(B_i, B_j) = \frac{1}{|B_i||B_j|} \sum_{p=1}^{|B_i|} \sum_{q=1}^{|B_j|} k^r(\mathbf{x}_{i,p}, \mathbf{x}_{j,q}) \tag{9}$$

This is basically the normalized set kernel for kernel k^r at instance level. The conclusion is then established by the equivalence between the set kernel and the proposed "label-mean" formulation, which takes the average of instance labels for bag prediction, as shown in the previous lemma. □

[1] The detailed steps are quite standard and similar to the derivation of dual of 2-norm SVMs, and are thus omitted here.

Theorem 1 establishes the validity of the "label-mean" formulation for MI classification. Intuitively, if positive and negative instances are linearly separable, we can build a SVM classifier based on the "label-mean" formulation with an order-r polynomial kernel to guarantee separability at bag-level. In the nonlinear case, if instances are separable with the Gaussian kernel, bags are separable with a Gaussian kernel with a larger scale parameter in the formulation.

The weight vector $\mathbf{w}$ of the SVM classifier can be expressed in terms of the dual variables α_i's from the solution of the dual formulation in Equation 8 by the KKT condition

$$\mathbf{w} = \sum_i \alpha_i y_i \frac{1}{n_i} \sum_{p=1}^{n_i} \phi(\mathbf{x}_{i,p}) \tag{10}$$

By substituting the equation into Equation 4, we have the following instance and bag-level classifiers for the proposed formulation

$$f(\mathbf{x}) = \sum_i \alpha_i y_i \frac{1}{n_i} \sum_{p=1}^{n_i} k(\mathbf{x}_{i,p}, \mathbf{x}) + b \tag{11}$$

$$\begin{aligned} F(B) &= \sum_i \alpha_i y_i k_{set}(B_i, B) + b \\ &= \frac{1}{|B|} \sum_{q=1}^{|B|} \sum_i \alpha_i y_i \frac{1}{n_i} \sum_{p=1}^{n_i} k(\mathbf{x}_{i,p}, \mathbf{x}_q) + b \end{aligned} \tag{12}$$

The complexity of the above prediction functions depends on the number of nonzero α_i's as well as the number of instances in each bag. A single nonzero dual variable would render all instances in the corresponding bag to be SVs. Thus the total number of SVs is given by

$$N_{sv} = \sum_{i=1,\alpha_i \neq 0}^{m} n_i \tag{13}$$

which is proportional to the number of instances in the training set. Moreover, to make a bag-level prediction, one needs to evaluate $f(\mathbf{x})$ for all instances in the bag and average the instance prediction values. This creates a more adverse situation for MI prediction as compared to the SI instance case with SVM. To make prediction faster, it is essential to have a model with fewer vectors in the expansion in Equation 11.

3 Sparse SVM for MI Classification

A straightforward approach to achieve a sparse prediction model is to fit the trained classifier with a new classifier with reduced complexity. The Reduced Set (RS) is one such method [10]. It approximates the weight vector in Equation 10 with the following new weight vector

$$\mathbf{w} = \sum_{k=1}^{N_{xv}} \beta_k \phi(\mathbf{z}_k) \tag{14}$$

where $\mathbf{z}_k$'s are Expansion Vectors (XV) [2] whose feature maps $\phi(\mathbf{z}_k)$'s form the basis of linear expansion for $\mathbf{w}$, and N_{xv} denotes the number of XVs. Let $\beta = [\beta_1, \ldots, \beta_{N_{xv}}]$ be the vector of expansion coefficients and $\mathbf{Z} = [\mathbf{z}_1^T, \ldots, \mathbf{z}_{N_{xv}}^T]^T$ the concatenation of XVs in a column vector. They can be solved by solving the following optimization problem

$$(\beta, \mathbf{Z}) = \arg\min_{\beta, \mathbf{Z}} \left\| \sum_{k=1}^{N_{xv}} \beta_k \phi(\mathbf{z}_k) - \sum_i \alpha_i y_i \frac{1}{n_i} \sum_{p=1}^{n_i} \phi(\mathbf{x}_{i,p}) \right\|^2 \tag{15}$$

It can be solved by a greedy approach developed in [10].

Despite the simplicity of RS, it is basically a postprocessing method used for fitting any trained model. It does not produce classifier models by itself. Moreover, RS simply minimizes fitting error in model learning without utilizing any discriminant information. This may not be desirable for low sparsity cases when the fitting errors are high.

Alternatively, a sparse model can be learned directly from the SVM formulation by imposing the L_1 norm on the coefficients due to its sparsity preserving property. This strategy was employed by MILES [5], a popular SVM method for MI classification. However, we are still confronted with the trade-off between sparsity and accuracy. This is because sparsity is not explicitly imposed on the model. Therefore for difficult problems, either sparsity is sacrificed for better accuracy, or the other way around.

3.1 Model

Based on the aforementioned concerns, we propose a direct approach for building sparse SVM models for MI classification based on the formulation presented in the previous section. This is achieved by adding an explicit constraint to control the complexity of linear expansion for the weight vector $\mathbf{w}$ in Equation 10. In this way, we can control the number of kernel evaluations involved in the computation of prediction functions in Equations 11 and 12. Specifically, we aim at approximating $\mathbf{w}$ with a reduced set of feature maps while maintaining the large margin of the SVM objective function. The new optimization problem can thus be formulated as

$$\min_{\mathbf{Z}, \mathbf{w}, b} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{m} \ell \left(y_i, \frac{1}{n_i} \sum_{p=1}^{n_i} (\mathbf{w}^T \phi(\mathbf{x}_{i,p}) + b) \right) \tag{16}$$

Intuitively, we can view the above formulation as searching for the optimal solution that minimizes the bag-level loss in a subspace spanned by $\phi(\mathbf{z}_k)$'s in the RKHS $\mathcal{H}$ induced by instance kernel k instead of the whole RKHS. By directly specifying the number of XVs N_{xv} in $\mathbf{w}$, the optimal solution is guaranteed to

[2] We have adopted the same terminology here as in [12] for the same reason. Technically, an XV, which can be an arbitrary point in the instance space, is different from an SV, which must be chosen from an existing instance in the training set.

reside in a subspace whose dimension is no larger than N_{xv}. The above formulation can be regarded as a joint optimization problem that toggles between searches for the optimal subspace and for the optimal solution in the subspace. With $N_{xv} \ll N_{sv}$, we can build a much sparser model for MI prediction.

Let $\mathbf{K_Z}$ denote the $N_{xv} \times N_{xv}$ Gram matrix for XVs $\mathbf{z}_k$'s, with the (i,j)th entry given by $k(\mathbf{z}_i, \mathbf{z}_j)$, $\mathbf{K}_{B_i,\mathbf{Z}}$ denote the $n_i \times N_{xv}$ Gram matrix between instances in bag i and XVs, with the (p,j)th entry given by $k(\mathbf{x}_{i,p}, \mathbf{z}_j)$, and $\mathbf{1}_{n_i}$ be the n_i dimensional column vector with value of 1 for each element. We can rewrite optimization problem in Equation 16 in terms of β by substituting Equation 14 into the cost function in Equation 16. This leads to the following objective function we solve for sparse SVM model for MI classification.

$$\min_{\beta,b,Z} Q(\beta, b, \mathbf{Z}) = \frac{1}{2}\beta^T \mathbf{K_Z} \beta + C \sum_i \ell \left(y_i, \frac{1}{n_i} \mathbf{1}_{n_i}^T \mathbf{K}_{B_i,\mathbf{Z}} \beta + b \right) \tag{17}$$

3.2 Optimization Strategy

The sparse SVM formulation in Equation 17 involves the joint optimization of two inter-dependent sets of target variables - the classifier weights (β, b) and XVs $\mathbf{z}_k$'s. Change in one of them would influence the optimal solution of the other. A straightforward strategy to tackle this problem is via alternating minimization [13]. However, alternating optimization lacks convergence guarantee and usually leads to slow convergence in practice [13, 14]. Hence, we adopt a more efficient and effective strategy to solve the sparse SVM model here by viewing it as an optimization problem for the optimal value function. Specifically, we convert the original problem defined in Equation 17 into the following problem which depends on variable $\mathbf{Z}$ only

$$\min_{\mathbf{Z}} g(\mathbf{Z}) \qquad \text{with} \qquad g(\mathbf{Z}) = \min_{\beta,b} Q(\beta, b, \mathbf{Z}) \tag{18}$$

$g(\mathbf{Z})$, the new objective function, is special in the sense that it is the optimal value of Q optimized over variables (β, b). The evaluation of $g(\mathbf{Z})$ at a fixed point $\mathbf{Z}$ is equivalent to training a 2-norm SVM given the XVs and computing the cost of the trained model in Equation 17. To train the 2-norm SVM, we minimize function $Q(\beta, b, \mathbf{Z})$ over β and b by fixing $\mathbf{Z}$. This can be done easily with various numerical optimization routines. In our work, we used the limited memory BFGS (L-BFGS) algorithm for its efficiency and super-linear convergence rate. The implementation of L-BFGS requires the cost $Q(\beta, b, \mathbf{Z})$ and the gradient information below

$$\frac{\partial Q}{\partial \beta} = \mathbf{K_Z}\beta + C \sum_{i=1}^{m} \frac{1}{n_i} \ell'_{f_i}(y_i, f_i) \mathbf{K}_{B_i,\mathbf{Z}}^T \mathbf{1}_{n_i} \tag{19}$$

$$\frac{\partial Q}{\partial b} = C \sum_{i=1}^{m} \frac{1}{n_i} \ell'_{f_i}(y_i, f_i) \tag{20}$$

where the partial derivative of the squared Hinge loss function ℓ with respect to f_i is given by

$$\ell'_{f_i}(y_i, f_i) = \begin{cases} 2(f_i - y_i) & y_i f_i < 1 \\ 0 & y_i f_i \geq 1 \end{cases} \tag{21}$$

Denote $\overline{\beta}$ and $\overline{b}$ as the minimizer of $Q(\beta, b, \mathbf{Z})$ at $\mathbf{Z}$, the value of function $g(\mathbf{Z})$ is then given by

$$g(\mathbf{Z}) = \frac{1}{2}\overline{\beta}^T \mathbf{K}_{\mathbf{Z}} \overline{\beta} + C \sum_i \ell \left(y_i, \frac{1}{n_i} \mathbf{1}_{n_i}^T \mathbf{K}_{B_i,\mathbf{Z}} \overline{\beta} + \overline{b} \right) \tag{22}$$

Note that we assume the Gram matrix $\mathbf{K}_{\mathbf{Z}}$ to be positive definite, which is always the case with the Gaussian kernel for distinct $\mathbf{z}_k$'s[3]. Q is then a strictly convex function with respect to β and b, and the optimal solution at each $\mathbf{Z}$ is unique. This makes $g(\mathbf{Z})$ a proper function with unique value for each $\mathbf{Z}$. Moreover, the uniqueness of optimal solution also makes it possible for the derivative analysis for $g(\mathbf{Z})$.

Existence and computation of the derivative of the optimal value function has been well studied in the optimization literature. Specifically, Theorem 4.1 of [15] has provided the sufficient conditions for the existence of derivative of $g(\mathbf{Z})$. According to the theorem, the differentiability of $g(\mathbf{Z})$ is guaranteed by the uniqueness of optimal solution $\overline{\beta}$ and $\overline{b}$ as we discussed earlier, and by the differentiability of $Q(\beta, b, \mathbf{Z})$ with respect to β and b, which is ensured by the square Hinge loss function we adopted. Moreover, the derivative of $g(\mathbf{Z})$ can be computed at each given $\mathbf{Z}$ by substituting the minimizers $\overline{\beta}$ and $\overline{b}$ into Equation 22 and taking the derivative in the following as if $g(\mathbf{Z})$ does not depend on $\overline{\beta}$ and $\overline{b}$

$$\begin{aligned} \frac{\partial g}{\partial \mathbf{z}_k} = & \sum_{i=1}^{N} \overline{\beta}_i \overline{\beta}_k \frac{\partial k(\mathbf{z}_i, \mathbf{z}_k)}{\partial \mathbf{z}_k} \\ & + C \sum_{i=1}^{m} \frac{1}{n_i} \ell'(y_i, f_i) \overline{\beta}_k \sum_{p=1}^{n_i} \frac{\partial k(\mathbf{x}_{i,p}, \mathbf{z}_k)}{\partial \mathbf{z}_k} \end{aligned} \tag{23}$$

The derivative terms in the above equation depend on the specific choice of kernels. In our work, we have adopted the following Gaussian kernel

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right)$$

where γ is the scale parameter for the kernel. Note that the power of a Gaussian kernel is still a Gaussian kernel. Hence, according to Theorem 2 in Section 2, if instances are separable with Gaussian kernel with scale parameter γ, then bags are separable with Gaussian kernel with scale parameter $r\gamma$ for some r

[3] In practice, we can enforce the positive definiteness by adding a small value to the diagonal of $\mathbf{K}_{\mathbf{Z}}$.

Algorithm 1. Sparse SVM for MI Classification

Input: data (B_i, y_i), N_{xv}, λ, $s_{\max}$ and $t_{\max}$
Output: classifier weights β, b and XVs $\mathbf{Z}$
Set $t = 0$ and initialize $\mathbf{Z}$
Solve $\min_{\beta,b} Q(\beta, b, \mathbf{Z}^{(t)})$ and denote the optimizer as $(\beta^{(t)}, b^{(t)})$ and the optimal value as $g(\mathbf{Z}^{(t)})$
repeat
  for $s = 1$ **to** $s_{\max}$ **do**
    Set $\mathbf{Z}^{'} = \mathbf{Z}^{(t)} - \lambda \frac{\partial g(\mathbf{Z})}{\partial \mathbf{Z}}$
    Solve $\min_{\beta,b} Q(\beta, b, \mathbf{Z}^{'})$ and denote the optimizer as $(\beta^{'}, b^{'})$ and the optimal value as $g(\mathbf{Z}^{'})$
    if $g(\mathbf{Z}^{'}) < g(\mathbf{Z}^{(t)})$ **then**
      Set $(\beta^{(t+1)}, b^{(t+1)}) = (\beta^{'}, b^{'})$, $\mathbf{Z}^{(t+1)} = \mathbf{Z}^{'}$
      Set $\lambda = 2\lambda$ if s equals 1
      break
    end if
    Set $\lambda = \lambda/2$
  end for
  Set $t = t + 1$
until Convergence or $t \geq t_{\max}$ or $s \geq s_{\max}$

using the "label-mean" formulation. Thus in practice, we only have to search over the γ parameter for Gaussian kernel. With the use of Gaussian kernel, the partial derivative terms in Equation 23 can be replaced by

$$\frac{\partial k(\mathbf{x}, \mathbf{z}_k)}{\partial \mathbf{z}_k} = 2\gamma(\mathbf{x} - \mathbf{z}_k)k(\mathbf{x}, \mathbf{z}_k)$$

3.3 Algorithm and Extension

With $g(\mathbf{Z})$ and its derivative given in Equations 22 and 23, we can develop a gradient descent approach to solve the overall optimization problem for sparse SVM in Equation 17. The detailed steps are outlined in Algorithm 1. Besides input data and N_{xv}, additional input parameters include λ, the initial step size, $s_{\max}$, the maximum number of line searches allowed, and $t_{\max}$, the maximum number of iterations. For line search, we implemented a strategy similar to backtracking, without imposing the condition on sufficient decrease. Any step size resulting in a decrease in function value is immediately accepted. This is more efficient than more sophisticated line search strategies with significant reduction in the number of expensive function evaluations.

The optimization scheme we used has also been adopted previously for sparse kernel machine [12] and simple Multiple Kernel Learning (MKL) [14]. The major difference here is that the SVM problem is solved directly in its primal formulation when evaluating the optimal value function $g(\mathbf{Z})$, whereas in [12] and [14], a dual formulation of SVM is solved instead by invoking standard SVM solvers.

This is mainly due to the use of squared Hinge loss in the cost function, which makes the primal formulation continuously differentiable with respect to classifier parameters. In contrast, both sparse kernel machine [12] and simple MKL [14] used non-differentiable Hinge loss for the basic SVM model, which has to be treated in the dual form to guarantee the differentiability of the optimal value function. Solving the primal problem has great advantage in computational complexity as compared to the dual. For our formulation, it is cheaper to solve the primal problem for SVM since it only involves $N_{xv}+1$ variables. In contrast, the complexity of the dual problem is much higher. It involves the computation and cache of the kernel matrix in solving the SVM. Moreover, the gradient evaluation with respect to the XVs is also much more costly, as it needs to aggregate over the gradient value for each entry in the kernel matrix, which is the sum of inner products between vectors of kernel function values over instances and XVs. The complexity for gradient computation scales with $O(N_{xv}N^2)$, where N is the total number of instances in the training set. This is much higher than the complexity of $O(N_{xv}N)$ in the primal case.

The proposed algorithm can also be extended to deal with multi-class MI problems. Since a multi-class problem can be decomposed into several binary problems using various decomposition schemes, each binary problem may introduce a different set of XVs by applying the binary version of algorithm directly. Therefore, the XVs have to be learned in a joint fashion for multi-class problems to ensure that each binary classifier shares the same XVs. Consider M binary problems and let β^c, b^c denote the weight and bias for the cth classifier respectively ($c = 1, \ldots, M$), the optimization problem is only slightly different from Equation 17

$$\begin{aligned} \min_{\beta,\mathbf{b},Z} Q((\beta, \mathbf{b}, \mathbf{Z}) &= \sum_{c=1}^{M} Q^c(\beta^c, b^c, \mathbf{Z}) \\ &= \sum_{c=1}^{M} \left(\frac{1}{2} {\beta^c}^T \mathbf{K}_{\mathbf{Z}} \beta^c + C \sum_i \ell \left(y_i^c, \frac{1}{n_i} \mathbf{1}_{n_i}^T \mathbf{K}_{B_i,\mathbf{Z}} \beta^c + b^c \right) \right) \end{aligned} \tag{24}$$

The same strategy can be adopted for the optimization of the above equation by introducing $g(\mathbf{Z})$ as the optimal value function of Q over β^c's and b's. The first step of each iteration is the same, only that M SVM classifiers are trained instead of one. These M classifiers can be trained separately by minimizing $Q^c(\beta^c, b^c, \mathbf{Z})$ for $c = 1, \ldots, M$. The argument on the existence of the derivative of $g(\mathbf{Z})$ is still true, which can be computed with a minor modification via

$$\begin{aligned} \frac{\partial g}{\partial \mathbf{z}_k} &= \sum_{c=1}^{M} \sum_{i=1}^{N_{xv}} \overline{\beta}_i^c \overline{\beta}_k^c \frac{\partial k(\mathbf{z}_i, \mathbf{z}_k)}{\partial \mathbf{z}_k} \\ &\quad + C \sum_{c=1}^{M} \sum_{i=1}^{m} \frac{1}{n_i} \ell'(y_i^c, f_i) \overline{\beta}_k^c \sum_{p=1}^{n_i} \frac{\partial k(\mathbf{x}_{i,p}, \mathbf{z}_k)}{\partial \mathbf{z}_k} \end{aligned} \tag{25}$$

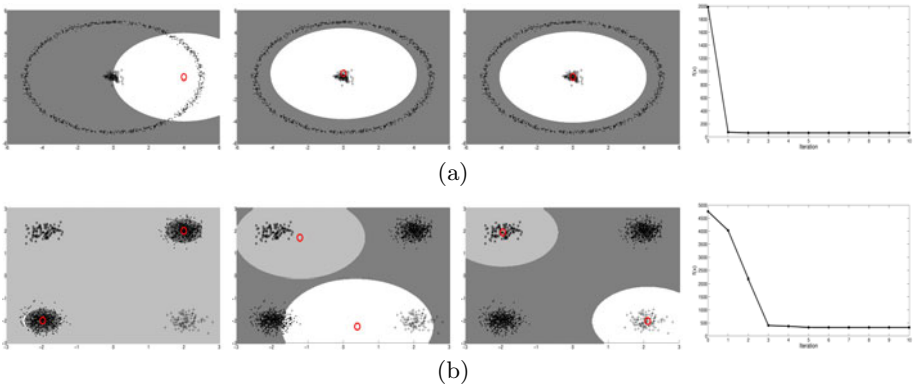


Fig. 1. Demonstration of the proposed sparse SVM classifier on synthetic (a) binary and (b) 3-class MI data sets

4 Experimental Results

4.1 Synthetic Data Examples

In our experiments, we first show two examples on synthetic data to demonstrate the interesting properties of the proposed sparse SVM algorithm for MI classification. Figure 1(a) shows an example of binary MI classification, where each positive bag contains at least one instance in the center, while each negative bag contains only instances on the ring. Figure 1(b) shows a MI classification problem with three classes. Instances are generated from four Gaussians $N(\mu_i, \sigma^2)(i = 1, \ldots, 4)$ with $\mu_1 = [-2, 2]^T$, $\mu_2 = [2, 2]^T$, $\mu_3 = [2, -2]^T$, $\mu_4 = [-2, -2]^T$, and $\sigma = 0.25$. Bags from class 3 contains only instances randomly drawn from $N(\mu_2, \sigma^2)$ and $N(\mu_4, \sigma^2)$, whereas each bag from class 1 contains at least one instance drawn from $N(\mu_1, \sigma^2)$ and each bag from class 2 contains at least one instance drawn from $N(\mu_3, \sigma^2)$. For the binary example, a single XV in the center is sufficient to discriminate between bags from two classes. For the 3-class example, two XVs are needed for discriminant purposes whose optimal locations should overlap with μ_1 and μ_3, the centers of class 1 and 2. For both cases, we initialize our algorithm with poor XV locations as shown by the circles on the first column of Figure 1. The next two columns show the data overlayed with XVs updated after the first and final iterations respectively. Based on the trained classifier, we can also partition the data space into predicted regions for each class. The partition is also shown in our plots and regions for different classes are described by different shades. It can be seen that, despite the poor initialization, the algorithm is able to find good XVs and make the right decision. The convergence is quite fast, with the first iteration already making a pronounced improvement over the initialization and XV

locations are refined over the remaining iterations. This is further demonstrated by the monotonically decreasing function values over iterations on the rightmost column of Figure 1, with significant decrease in the first few iterations.

4.2 Results on Real-World Data

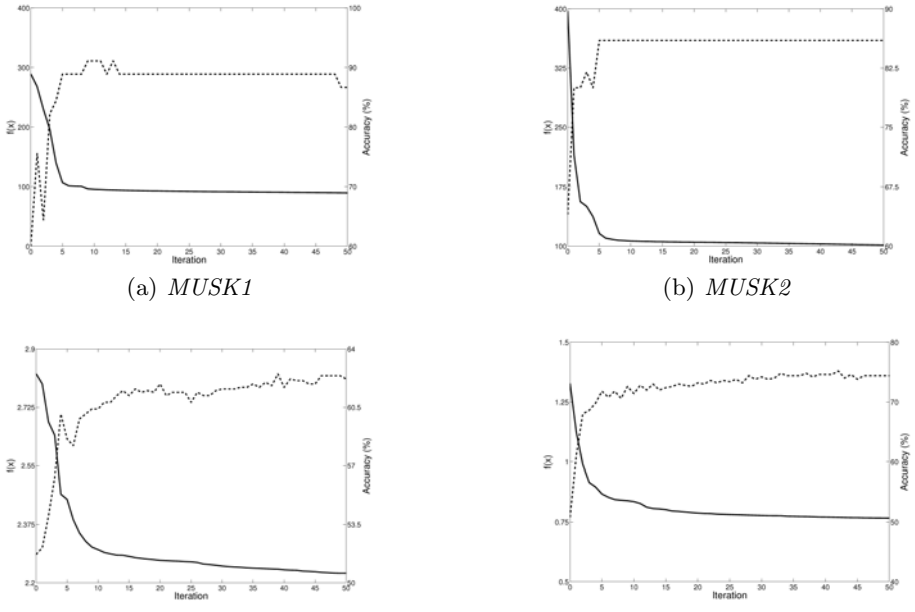
We now turn our attention to real-world data. Five data sets were used in our experiment. These include two data sets on drug activity prediction (*MUSK1* and *MUSK2*), two data sets on image categorization (*COREL-10* and *COREL-20*) and one data set on music genre classification (*GENRE*). *MUSK1* and *MUSK2* were first introduced in [1] and have been widely used as the benchmark for MI classification. *MUSK1* contains 47 positive bags and 45 negative ones, with an average of 5.2 instances in each bag. *MUSK2* contains 39 positive bags and 63 negative ones, with an average of 64.7 instances in each bag. *COREL-10* and *COREL-20* were from the COREL image collection and introduced for MI classification in [5]. *COREL-20* contains 2000 JPEG images from 20 categories, with 100 images per category. *COREL-10* is a subset of *COREL-20* with images from the first 10 categories. Each image is segmented into $2-13$ regions from which color and texture features are extracted. For our experiment, we simply used the processed features obtained from the author's website. *GENRE* was originally introduced in [16] and is first tested for MI classification in this paper. The data set contains 1000 audio tracks equally distributed in 10 music genres. We split each track, which is roughly 30 seconds long, into overlapping 3-second segments with 1-second steps. Audio features are then extracted from each segment following the procedures in [16].

We have compared the proposed sparse SVM for MI classification (SparseMI) with existing SVM implementations for MI classification, including MI-SVM [3],mi-SVM [3], MI-kernel [2] and MILES [5], as well as two alternative naive approaches to sparse MI classification - RS and RSVM. RS is discussed in the beginning of Section 3, and RSVM is a reduced SVM classifier learned from a randomly selected set of XVs [9], which is equivalent to running SparseMI without any optimization. All three methods can control the sparsity of the prediction model explicitly, so we have tested them with varied sparsity by ranging from 10, 50 up to 100 XVs. For *MUSK1* and *MUSK2*, we performed 10-fold Cross Validation (CV) and recorded CV accuracies. For other data sets, we have randomly selected 50% of the data for training and the remaining 50% for testing and recorded test accuracies. The experiments were repeated 10 times for each data set with different random partitions. Instance features are normalized to zero mean and unit standard deviation for all data sets. The Gaussian kernel was adopted for all SVM models being compared. Kernel parameter γ and regularization parameter C were chosen via 3-fold cross validation on the training set. For SparseMI, we set $s_{\max} = 10$, $t_{\max} = 50$ and λ equal to the average pairwise distance between initial XVs. Test results for different methods used in our experiment are reported in Table 1.

Table 1. Performance comparison for different methods on MI classification. Results are organized in four blocks. The first block shows the mean accuracies in percentage (first row for each method) as well as number of SVs (second row for each method) obtained by existing SVM methods. The next three blocks show the mean accuracies for three sparse SVM models with given number of XVs N_{xv}. For each N_{xv} value (i.e. block), the highest accuracy for each data set (i.e. column) is highlighted. More than one accuracy values may be highlighted in each column if the methods achieve equal performances according to the results of significance tests on the accuracy values over 10 test runs.

Data set		*MUSK1*	*MUSK2*	*COREL10*	*COREL20*	*GENRE*
mi-SVM		87.30 400.2	80.52 2029.2	75.62 1458.2	52.15 2783.3	80.28 12439
MI-SVM		77.10 277.1	83.20 583.4	74.35 977	55.37 2300.1	72.48 3639.8
MI-Kernel		89.93 362.4	90.26 3501	84.30 1692.6	73.19 3612.7	77.05 13844
MILES		85.73 40.8	87.64 42.5	82.86 379.1	69.16 868	69.87 1127.9
$N_{xv} = 10$	RS	**88.68**	86.39	75.13	55.20	54.68
	RSVM	74.66	77.70	69.25	48.53	46.01
	SparseMI	**88.44**	**88.52**	**80.10**	**62.49**	**71.28**
$N_{xv} = 50$	RS	**89.61**	**89.92**	79.87	66.27	65.27
	RSVM	87.23	86.71	76.94	62.54	63.58
	SparseMI	**89.98**	88.02	**84.19**	**71.66**	**75.28**
$N_{xv} = 100$	RS	**90.18**	**89.16**	78.81	65.35	67.77
	RSVM	89.02	88.26	77.63	63.82	67.41
	SparseMI	**90.40**	87.98	**84.31**	**72.22**	**76.06**

From the results, we can see that the proposed SparseMI is quite promising with the optimal balance between performance and sparsity. Compared to existing SVM methods like MI-kernel and MILES, SparseMI has comparable accuracy rates yet with significantly fewer XVs in the prediction function. Compared to alternative sparse implementations like RS and RSVM, SparseMI achieves better performances in majority of the cases. The gap in performance is more evident with smaller number of XVs, as can be observed for the case of $N_{xv} = 10$. With increasing value of N_{xv}, the performance of SparseMI is further improved. For $N_{xv} = 100$, SparseMI performs comparably with MI-Kernel, which can be regarded as the dense version of SparseMI. As a result, MI-Kernel produces more complex prediction models with far more SVs than SparseMI. Compared with MILES, which has better sparsity than other SVM methods, SparseMI not only achieves better performance but obtains sparser models. This is especially true in the cases of multiclass MI classification, for reasons we have discussed in the previous section. Moreover, with SparseMI, we can explicitly control its sparsity by specifying the number of XVs, while the sparsity of MILES can only be specified implicitly with the regularization parameter.



(a) *MUSK1* (b) *MUSK2*

(c) *COREL-20* (d) *GENRE*

Fig. 2. Stepwise demonstration of SparseMI optimization algorithm with plots of cost function values (in solid lines) and test accuracies (broken lines) over each iteration

To further demonstrate the effectiveness of SparseMI in optimizing the XVs over iterations, we show some stepwise examples in Figure 2. For each data set (except COREL-10 which behaves similarly to COREL-20), we plotted the cost function value and test accuracy over each iteration for a $50\% - 50\%$ random partitioning of the training and testing sets. It can be seen clearly that the cost function value monotonically decreases with each iteration, accompanied by the tendency of improvement for the test accuracy over the iterations. The decrease of cost function value and improvement of test accuracy are especially obvious for the first few iterations. After about 10 iterations, the test accuracy becomes gradually stable and does not change too much over further iterations despite small perturbations.

5 Conclusions

In this paper, we have proposed the first explicit formulation for sparse SVM learning in MI classification. Our formulation is descended from sparse kernel machines [12] and builds on top of an equivalent formulation of MI kernel [2]. Unlike MI kernel, the proposed technique can produce a sparse prediction model with controlled complexity while still performing competitively compared to MI kernel and other non-sparse methods.

Acknowledgment. This works is supported by the Australian Research Council under the Discovery Project (DP0986052) entitled "Automatic music feature extraction, classification and annotation".

References

1. Dietterich, T.G., Lathrop, R.H., Lozano-perez, T.: Solving the multiple-instance problem with axis-parallel rectangles. Artificial Intelligence 89, 31–71 (1997)
2. Gartner, T., Flach, A., Kowalczyk, A., Smola, A.J.: Multi-instance kernels. In: Intl. Conf. Machine Learning, pp. 179–186 (2002)
3. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Advances in Neural Information Processing Systems, pp. 561–568 (2003)
4. Cheung, P.M., Kwok, J.T.Y.: A regularization framework for multiple-instance learning. In: Intl. Conf. Machine Learning (2006)
5. Chen, Y., Bi, J., Wang, J.: Miles: Multiple-instance learning via embedded instance selection. IEEE Trans. on Pattern Analysis and Machine Intelligence 28(12), 1931–1947 (2006)
6. Zhou, Z.H., Xu, J.M.: On the relation between multi-instance learning and semi-supervised learning. In: Intl. Conf. Machine Learning (2007)
7. Li, Y.F., Kwok, J.T., Tsang, I., Zhou, Z.H.: A convex method for locating regions of interest with multi-instance learning. In: European Conference on Machine Learning (2009)
8. Smola, A.J., Scholkopf, B.: Sparse greedy matrix approximation for machine learning. In: Intl. Conf. Machine Learning, pp. 911–918 (2000)
9. Lee, Y.J., Mangasarian, O.L.: Rsvm: Reduced support vector machines. In: SIAM Conf. Data Mining (2001)
10. Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press, Cambridge (2002)
11. Keerthi, S., Chapelle, O., DeCoste, D.: Building support vector machines with reduced classifier complexity. Journal of Machine Learning Research 7, 1493–1515 (2006)
12. Wu, M., Scholkopf, B., Bakir, G.: A direct method for building sparse kernel learning algorithms. Journal of Machine Learning Research 7, 603–624 (2006)
13. Bezdek, J.C., Harthaway, R.J.: Convergence of alternating optimization. Journal Neural, Parallel & Scientific Computations 11(4) (2003)
14. Rakotomamonjy, A., Bach, F.R., Canu, S., Grandvalet, Y.: Simplemkl. Journal of Machine Learning Research 9, 2491–2521 (2008)
15. Bonnans, J.F., Shapiro, A.: Optimization problems with pertubation: A guided tour. SIAM Review 40(2), 202–227 (1998)
16. Tzanetakis, G., Cook, P.: Music genre classification of audio signals. IEEE Trans. Speech and Audio Processing 10(5), 293–302 (2002)