Fast Bundle Algorithm for Multiple-Instance Learning

Charles Bergeron, *Member*, *IEEE*, Gregory Moore, *Member*, *IEEE*, Jed Zaretzki, Curt M. Breneman, and Kristin P. Bennett

Abstract—We present a bundle algorithm for multiple-instance classification and ranking. These frameworks yield improved models on many problems possessing special structure. Multiple-instance loss functions are typically nonsmooth and nonconvex, and current algorithms convert these to smooth nonconvex optimization problems that are solved iteratively. Inspired by the latest linear-time subgradient-based methods for support vector machines, we optimize the objective directly using a nonconvex bundle method. Computational results show this method is linearly scalable, while not sacrificing generalization accuracy, permitting modeling on new and larger data sets in computational chemistry and other applications. This new implementation facilitates modeling with kernels.

Index Terms—Artificial intelligence, machine learning, nonsmooth optimization, bundle methods, multiple-instance learning, ranking, medicine and science.

1 Introduction

Multiple-instance learning (MIL) is a variation of supervised learning where the output information is only known for bags of items (or instances), as opposed to for each item. Since first introduced [1], many MIL formulations have been proposed. Multiple-instance classification (MIC) is the more common model, with applications such as drug discovery [1], image annotation [2], medical diagnosis [3], and hard-drive failure prediction [4]. The support vector machine (SVM) has been generalized to MIC [2], [3], [5]. Algorithms for MIC [6], [7], [8], [9], [10], [11], [12] also include from diverse density [13], [14] and neural networks [15]. Other multiple-instance regression methods exist, but are less widespread [16], [17]; a recent algorithm for multiple-instance outlier detection has been proposed [18].

We focus on challenging MIL ranking problems, recently developed for image and chemistry applications [19], [20]. Under multiple-instance ranking (MIRank), bags of items are ordered with respect to each other (see Fig. 1d). In the image retrieval application [19], images (bags) are ranked with features computed for the segments (items). In our preliminary work, we introduced MIRank to predict the experimental sites of metabolic liability in drug-like compounds. These problems have three levels of structure: Items (atoms) belong to bags (potential sites of metabolic liability) and bags belong to boxes (compounds). In this problem, the partial ranking of bags within each box is known. The task is to find a ranking function that consistently identifies the *preferred* bag in each box (Fig. 1e). The other bags in a box are called *undesirable*. As always in MIL, there exists the ambiguity as to which item or items in a bag results in this bag being preferred over the others in a box. The features describe the atoms in each compound. We previously showed that a MIRank paradigm produces models with better predictive ability than when problems are approximated by the simpler MIC framework [20]. This demonstration is not repeated in this paper. Classification, ranking, and MIL tasks are compared in Fig. 1.

Classic classification, regression, or ranking SVM formulations are formulated as a constrained linear or convex quadratic (LP or QP) problem using the hinge loss, and then converted to a constrained smooth convex LP or QP by adding slack variables. The size of this LP or QP grows with sample size, and so this approach is not scalable. However, directly optimizing the nonsmooth convex loss function using various subgradient-based methods, including cutting-plane and bundle methods, achieves linearly scalable primal SVM methods [21], [22]. An additional difficulty appears with MIL models: The objective is not convex. Prior efforts [2], [3], [5], [19], [20] formulate the problem as a nonsmooth nonconvex hinge loss, and then convert the problem to an equivalent smooth problem by adding slack variables. The nonconvexity in the loss is handled through iterative solution of subproblems that are convex. These SVM-like MIL algorithms [2], [3], [5], [19], [20] have limited scalability, hindering the modeling of large data sets.

The paper establishes that a recent state-of-the-art but little-known general-purpose nonsmooth nonconvex bundle method [23] from the mathematical programming

- *C. Bergeron is with the Departments of Mathematical Sciences and Electrical, Systems, and Computer Engineering, Rensselaer Polytechnic Institute, 110 Eighth Street, Troy, NY 12180. E-mail: chbergeron@gmail.com.*
- *G. Moore is with the Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: mooregm@gmail.com.*
- *J. Zaretzki and C.M. Breneman are with the Department of Chemistry and Chemical Biology, Rensselaer Polytechnic Institute, 110 Eighth Street, Troy, NY 12180. E-mail: zaretj@gmail.com, brenec@rpi.edu.*
- *K.P. Bennett is with the Departments of Mathematical Sciences and Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: bennek@rpi.edu.*

Manuscript received 10 May 2011; accepted 14 Aug. 2011. online 6 Oct. 2011.
Recommended for acceptance by M. Meila.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2011-05-0307.
Digital Object Identifier no. 10.1109/TPAMI.2011.194.

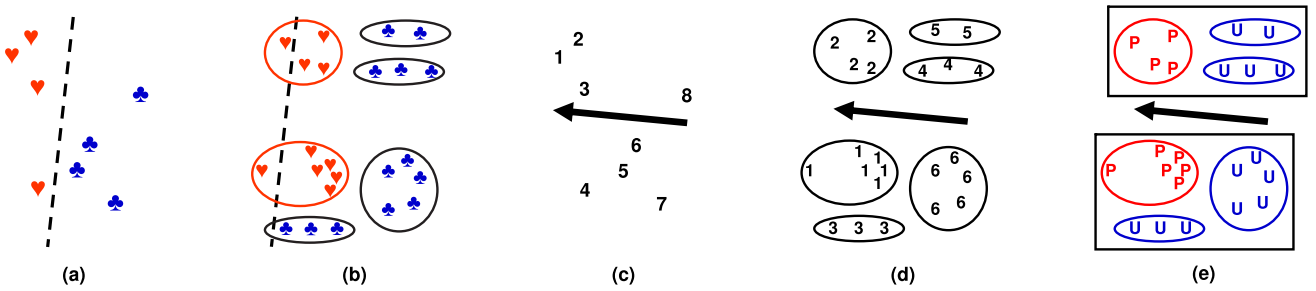


Fig. 1. Machine learning paradigms. Classification attempts to find a separator between items from two classes. (a) The decision curve (dashed line) divides the hearts from the clubs. This paper treats the multiple-instance classification problem, where bags (ellipses) of items are classified. (b) Bags having at least one item to the left of the decision curve are classified as hearts. All items in a bag classified as clubs must be to the right of the decision curve. Under ranking, the ranking function (arrow) orders items (represented as numbers that indicate rank ordering) as in (c). Under multiple-instance ranking, bags (ellipses) are ranked. The highest-ranked item determines the ranking for its whole bag, as in (d). This paper deals with the special case where rank ordering is only known within clusters of bags called boxes (rectangles). (e) Bags in each box are labeled *preferred* (P) or *undesirable* (U). The ranking function learns, for each box, the preferred bag.

community can achieve massive gains in speed over prior approaches on the computationally challenging nonconvex MIL problems. Simple subgradient algorithms are typically ineffective for nonconvex losses because subgradients do not necessarily correspond to directions of decrease. The effective implementation of the nonsmooth nonconvex bundle method for MIL is nontrivial and time consuming. Our algorithm, $\text{MIL}^{\text{bundle}}$, exploits recent advances in bundle methods [23] that have been effective on transductive SVM [24].

Our $\text{MIL}^{\text{bundle}}$ algorithm converges to local minima in linear time in the sample size without sacrificing generalization ability. Results on new data sets provided by computational chemists are reported, as well as on standard MIL data sets. The massive gain in speed afforded by $\text{MIRank}^{\text{bundle}}$ permits quick calculation of models on large data sets, a necessary step toward making useful contributions to drug design using computers.

This paper is outlined as follows: Section 2 motivates this work. Section 3 defines notation. Section 4 describes the MIL loss functions, while Section 5 specifies the bundle method used in this project. Section 6 presents the data sets used in this work. Section 7 provides implementation details, while Sections 8-10 provide experimental design and results for the experiments performed. Finally, Section 11 concludes with discussion and directions for future work.

2 Motivation

In this section, MIRank is introduced from the perspective of the drug design application that motivated its formulation.

Drug candidates with high potency must satisfy many additional criteria before they are considered safe and effective, including an assessment of their metabolites and lifetime in the body. Most of these considerations are governed by their metabolism by cytochrome P450 enzymes. The ability to make predictions about metabolism behavior in a drug candidate is an important facet of pharmaceutical research.

The ability of a drug administered orally to reach the bloodstream is called *bioavailability*. This is an important quality in drug design, being a convenient and inexpensive means of drug administration. In the stomach, a pill is broken down to the compound level, and these compounds proceed to the liver. Successful drugs cross the hepatic lining and reach the bloodstream without being degraded by cytochrome P450 isozymes so that their medicinal effect may be felt. The P450 cytochromes account for at least 75 percent of the total metabolic liability [25]. We consider the following nine P450 isozymes: CYPs 1A2, 2A6, 2B6, 2C8, 2C9, 2C19, 2D6, 2E1, and 3A4.

A data set of compounds that are degraded by a CYP isozyme is organized as follows: Each compound consists of usually one but possibly several potential site(s) of metabolic liability. A site usually consists of several atoms. The carbons and hydrogens forming one or several CH_3 groups that are topologically equivalent may constitute a site. Some sites possess a single atom, such as a sulfur. One or several sites are designated as the experimental sites of metabolic liability, where degradation is observed in a laboratory experiment. The response is therefore assigned at the site level. However, computational chemists calculate features for each atom in a compound. Data sets of this type are peculiar in two ways: their multiple-instance nature and their ranking nature. We discuss each in turn.

First, this is a multiple-instance problem because the response and the features are known at different levels of structure: Features are computed for the atoms (items) and the response is known for the sites (bags). Further, there exists an ambiguity as to which item in each bag determines the response for each bag, as is frequently the case in MIL. Second, this is a ranking problem within each box. For each compound (box), partial ranking information is known: One (or occasionally several) site(s) ranks higher than the others: These are called *preferred bags* and *undesirable bags*, respectively. The relative ranking among preferred bags or undesirable bags is unknown, and no ranking information is known across boxes.

We call this partial ranking within each box in a multiple-instance setting *multiple-instance ranking* or MIRank. Many new MIRank problems may be identified with the following template: For each *box*, find the *bag* that contains the best *item*. Following are some examples:

1. For each country, predict the city with the most profitable franchise.
2. For each document, predict the passage that contains the most pertinent phrase.

3. For each region, predict the subregion with the highest census statistic.
4. For each compound, predict the metabolic liability site whose atoms are involved in degradation.
5. For each sporting event, predict the nationality of the winning athlete.

Compare task 5 with the following:

5b. For each sporting event, predict the winning athlete.

These are not the same tasks, as the bag level (nationalities) is eliminated from the response structure for task 5b. Errors are treated differently under each. Consider the following example: The nation of Beeland is represented by cyclists Bao, Benji, and Bjork in a race. A MIRank model may incorrectly predict that Bao will win the gold medal for Beeland when in fact it is Benji that wins the race, but since task 5 involves predicting the nationality and not the individual cyclist, there is no prediction error. This ambiguity in MIL modeling is taken one step further in the drug design application, where it may be known that a hydrogen atom (item) gets abstracted from the experimental site of metabolic liability (preferred bag) in a given compound (box) under the effect of an isozyme, but not which one.

This paper aims to solve MIL problems, and in particular, to formulate and implement a scalable algorithm to find their solutions.

3 Notation and Definitions

The set of real numbers is denoted $\mathbb{R}$. Let $\boldsymbol{x} \in \mathbb{R}^n$ denote a vector and X, a matrix. The transpose of $\boldsymbol{x}$ is written $\boldsymbol{x}^T$. A vector of ones of appropriate length is denoted by $\boldsymbol{e}$. The family of vector p-norms for $p \geq 1$ is defined as $\|\boldsymbol{x}\|_p = p\sqrt{\sum_{i=1}^{n} |\boldsymbol{x}_i|^p}$. The root mean square (RMS) of a vector is $\boldsymbol{x}_{rms} = \sqrt{\frac{1}{n}\boldsymbol{x}^T\boldsymbol{x}}$. Let $\boldsymbol{x}^\star \in \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$ be an element in the set of optimal solutions to the mathematical problem $\min_{\boldsymbol{x}} f(\boldsymbol{x})$. The convex hull defined over set $\mathcal{X}$ of vectors is denoted $\mathrm{conv}(\mathcal{X})$.

A nonsmooth function is locally Lipschitz function, meaning that f is Lipschitz on every bounded set and is differentiable almost everywhere. The subdifferential of f at $\bar{\boldsymbol{x}}$ is a set [26], [27]:

$$\partial f(\bar{\boldsymbol{x}}) = \mathrm{conv}\,(\{\boldsymbol{g} \in \mathbb{R}^n | \boldsymbol{x} \in \mathbb{R}^n, \nabla f(\boldsymbol{x}) \rightarrow \boldsymbol{g}, \boldsymbol{x} \rightarrow \bar{\boldsymbol{x}}, \boldsymbol{x} \notin \Omega\}), \tag{1}$$

with Ω being the set of points where f is not differentiable. A subgradient $\boldsymbol{g}$ of f at $\bar{x}$ is any element of $\partial f(\bar{\boldsymbol{x}})$. This definition implies that it is possible to calculate the function value and a subgradient vector for all $\boldsymbol{x}$.

Consider functions $f(x)$ and $g(x)$. The notation $f(x) = \mathcal{O}(g(x))$ signifies that there exists a positive real number C and point a such that $|f(x)| \leq C|g(x)|$ for $x \geq a$ [28]. The computational complexity of a calculation on ℓ samples is linearly scalable (or simply *scalable*) if it is $\mathcal{O}(\ell)$.

4 MIL Loss Functions

This section examines loss functions for MIRank and MIC frameworks.

4.1 MIRank

For our applications, MIRank is applied in a partial ranking context, where some ranking relationships are known within clusters of bags called boxes. Boxes also come into play when computing prediction accuracy. The context for Hu et al.'s work [19] involved full ranking information between bags. As previously done [20], we formulate our MIRank loss for the general case, that is, across all known ordinal pairings. These are pairs (I, J) such that bag I ranks higher than bag J; I and J are also lists of indices referring to their items. Let $X \in \mathbb{R}^{\ell \times n}$ be the $\ell \times n$ data matrix having ℓ rows and n features. Let $\boldsymbol{x}_i$ denote a vector of features corresponding to item i in bag I and similarly let $\boldsymbol{x}_j$ denote item j in bag J. These are found as the appropriate rows of X. Further, let f denote the ranking function. A preferred bag ranks higher than an undesirable one, resulting in the following key inequality:

$$\max_{i \in I} f(\boldsymbol{x}_i) > \max_{j \in J} f(\boldsymbol{x}_j). \tag{2}$$

Assuming a linear model of the form $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{w}$ with weights $\boldsymbol{w} \in \mathbb{R}^n$, this inequality is rewritten as

$$\max_{i \in I} \boldsymbol{x}_i^T\boldsymbol{w} > \max_{j \in J} \boldsymbol{x}_j^T\boldsymbol{w}. \tag{3}$$

Different losses emerge at this stage, depending on strategies used to handle the $\max$ operators in (3). For example, we previously replaced the left hand side $\max$ with a convex combination of items i in bag I [20], following the approach of [5]. The resulting bilinear optimization problem is solved by alternatively solving two LP subproblems until convergence is observed. See Bergeron et al. [20] for details on this bilinear MIRank algorithm (or Mangasarian and Wild [5] for a similar treatment with respect to a MIC loss). As the number of variables and constraints in both subproblems grows linearly with sample size, this approach cannot be scalable.

This paper deals with inequality (3) differently. Instead of introducing slack variables to handle the $\max$ operators, they are dealt with directly. The items possessing the greatest function value in bags I and J are $i^\star$ and $j^\star$, respectively, with

$$i^\star = \arg\max_{i \in I} \left[\boldsymbol{x}_i^T\boldsymbol{w}\right], \tag{4}$$

$$j^\star = \arg\max_{j \in J} \left[\boldsymbol{x}_j^T\boldsymbol{w}\right]. \tag{5}$$

When several items achieve the maximum of (4), any convex combination of these points $\{\boldsymbol{x}_i\}$ is admissible as $\boldsymbol{x}_{i^\star}$; the same applies for $\boldsymbol{x}_{j^\star}$. We once again choose the hinge loss so that the empirical risk contribution for pair (I, J) is

$$\zeta_{I,J} = \max\left[0, 1 - \boldsymbol{x}_{i^\star}^T\boldsymbol{w} + \boldsymbol{x}_{j^\star}^T\boldsymbol{w}\right]. \tag{6}$$

Both $\boldsymbol{x}_{i^\star}^T\boldsymbol{w}$ and $\boldsymbol{x}_{i^\star}^T\boldsymbol{w}$ are convex, but the former is subtracted, introducing nonconvexity to this formulation. Summing over all pairs (I, J) of bags and inserting a structural risk term and tradeoff hyperparameter ν, the final loss function is

$$f_{MIRank}(\boldsymbol{w}) = \frac{1}{p}\|\boldsymbol{w}\|_p^p + \frac{\nu}{q}\sum_{\forall (I,J)} \zeta_{I,J}^q. \tag{7}$$

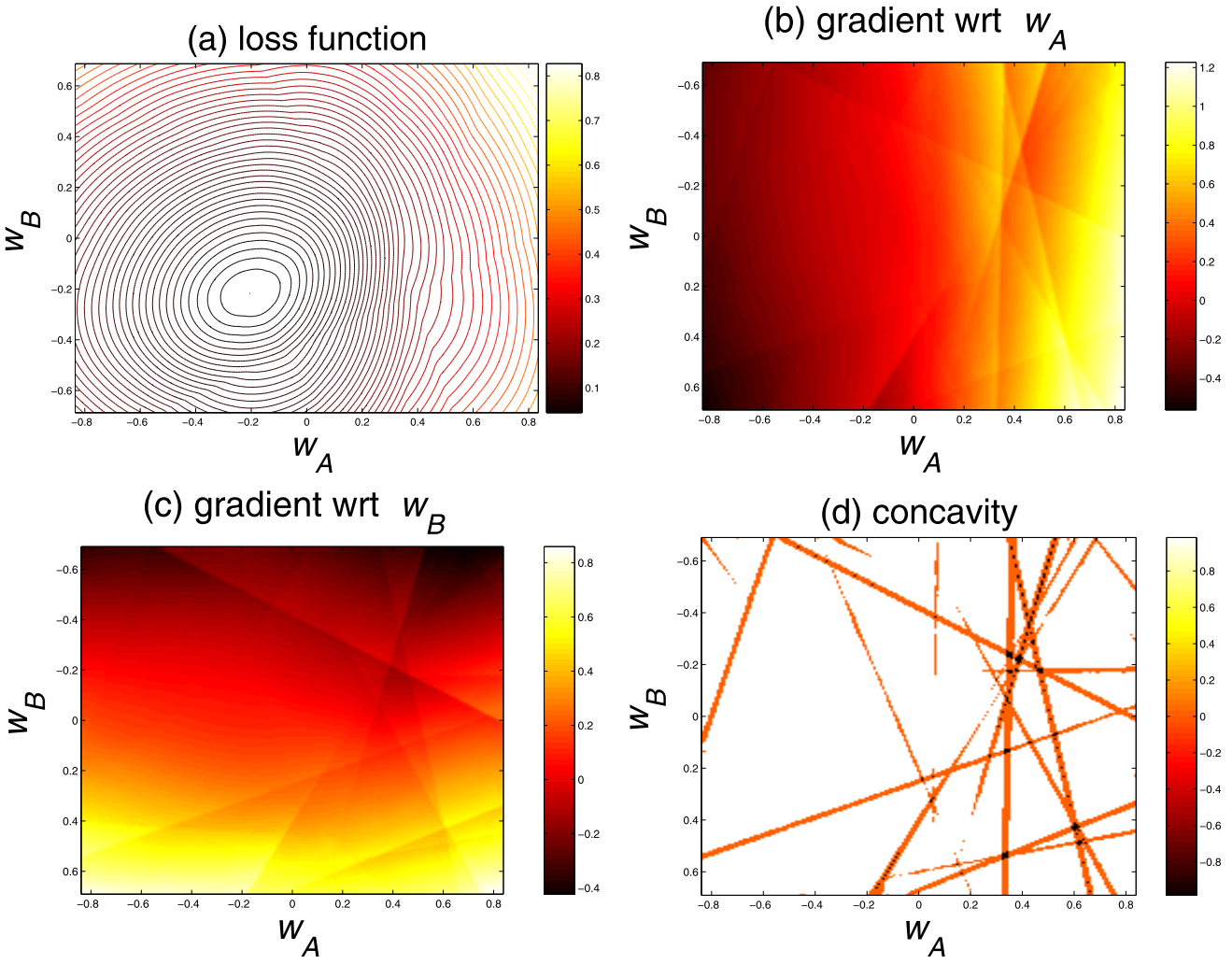


Fig. 2. Nonsmoothness and nonconvexity of MIRank loss. This figure was generated using the $\mathrm{CYP2B6}_{2009}$ data set discussed later in this paper. Each point on panels (a)-(d) demonstrates the effect on models generated from two features denoted A and B. (a) The MIRank loss is shown as a contour plot. The numerical gradients of the loss with respect to (b) w_A and (c) w_B are shown as heat maps—ridges in these indicate the presence of nonsmoothness. For each point, the numerical Hessian matrix is computed. (d) Positive-semidefinite areas appear in white, indefinite areas appear in orange, and negative-semidefinite areas appear in black. Nonwhite areas indicate nonconvexity.

In words, this loss penalizes structural risk (to control overfitting) and empirical risk stemming from ranking all pairs of bags across a data set for which the ordinal ranking is known (for our data sets, each combination of a preferred and undesirable bag within each box) (denoted by $\forall(I, J)$ in the summation).

Parameters p and q in loss (7) determine how structural and empirical risks are penalized, constituting an aspect of model selection. All $\max$ operators in (4)-(6) introduce kinks, making the loss nonsmooth. This nonconvex and nonsmooth behavior is graphed in Fig. 2.

The optimal model $\boldsymbol{w}$ is found as the solution of the following nonsmooth nonconvex optimization problem,

$$\min_{\boldsymbol{w}} \quad f_{MIRank}(\boldsymbol{w}), \tag{8}$$

using a specialized solver developed in the next section of this paper. This solver requires a subgradient for this loss. A subgradient $\boldsymbol{g}_{MIRank} \in \partial f_{MIRank}(\boldsymbol{w})$ when $p = q = 2$ is

$$\boldsymbol{g}_{MIRank} = \boldsymbol{w} + \nu \sum_{\forall(I,J)} (-\boldsymbol{x}_{i^\star} + \boldsymbol{x}_{j^\star})\zeta_{I,J}. \tag{9}$$

4.2 MIC

For MIC, the formulation works similarly, using (4). The empirical risk contributions are

$$\eta_I = \max[0, 1 - \boldsymbol{x}_{i^\star}^T \boldsymbol{w} - b], \tag{10}$$

$$\xi_J = \max[0, 1 + \boldsymbol{x}_{j^\star}^T \boldsymbol{w} + b], \tag{11}$$

for bags I and J labeled positive and negative. The loss is:

$$f_{MIC}(\boldsymbol{w}, b) = \frac{1}{p}\|\boldsymbol{w}\|_p^p + \frac{\nu}{q}\sum_{\forall I} \eta_I^q + \frac{\nu}{q}\sum_{\forall J} \xi_J^q. \tag{12}$$

In words, this loss penalizes structural risk, empirical risk for all bags belonging to the positive class (denoted by $\forall I$ in the summation) and empirical risk for all bags belonging to the negative class (denoted by $\forall J$ in the summation).

Loss (12) is nonconvex because $i^\star$ and $j^\star$ are themselves maximum values (4)-(5), and nonsmooth because of all $\max$ operators in (4)-(5), (10)-(11). The optimal model $\boldsymbol{w}$ is found as the solution of the following nonsmooth nonconvex optimization problem:

$$\min_{\boldsymbol{w},b} \quad f_{MIC}(\boldsymbol{w},b). \tag{13}$$

A subgradient $\boldsymbol{g}_{MIC} \in \partial f_{MIC}(\boldsymbol{w},b)$ when $p=q=2$ is

$$\boldsymbol{g}_{MIC} = \begin{bmatrix} \boldsymbol{w}3 - \nu \sum_{\forall I} \boldsymbol{x}_{i^\star} \eta_I + \nu \sum_{\forall J} \boldsymbol{x}_{j^\star} \xi_J \\ -\nu \sum_{\forall I} \eta_I + \nu \sum_{\forall J} \xi_J \end{bmatrix}.$$

4.3 Nonlinear Modeling with Kernels

The MIC and MIRank losses are extended to nonlinear MIL through the use of kernels functions. Kernels exploit duality to achieve nonlinear models (in a high-dimensional and possibly infinite-dimensional space) that is computed in a space whose dimensionality is equal to sample size [29]. In this section, we write the MIRank kernel loss for $p=2$; the MIC kernel loss is analogous.

Denote the dual model weights as ℓ-vector $\boldsymbol{\alpha}$. Vector $\boldsymbol{k}_i = k(X, \boldsymbol{x}_i)$ is the result of applying a valid kernel function k to data matrix X and item $\boldsymbol{x}_i$, and similarly for $\boldsymbol{k}_j$. Then (4)-(5) are rewritten

$$i^\star = \arg\max_{i \in I} \left[\boldsymbol{k}_i^T \boldsymbol{\alpha}\right], \tag{14}$$

$$j^\star = \arg\max_{j \in J} \left[\boldsymbol{k}_j^T \boldsymbol{\alpha}\right]. \tag{15}$$

The empirical risk contribution for pair (I,J) is

$$\zeta_{I,J} = \max\left[0, 1 - \boldsymbol{k}_{i^\star}^T \boldsymbol{\alpha} + \boldsymbol{k}_{j^\star}^T \boldsymbol{\alpha}\right]. \tag{16}$$

The kernel loss is therefore,

$$f_{MIRank}(\boldsymbol{\alpha}) = \frac{1}{2} \boldsymbol{\alpha}^T K \boldsymbol{\alpha} + \frac{\nu}{q} \sum_{\forall (I,J)} \zeta_{I,J}^q. \tag{17}$$

In order to limit the number of samples spanning the new space, we used a reduced kernel strategy [30], $K_{m,n} = k(\boldsymbol{x}_m, \boldsymbol{x}_n)$ with training samples $m \in \mathcal{M}$ and kernel columns $\mathcal{N} \subseteq \mathcal{M}$ [5]. This corresponds to training in a subspace of the kernel space defined by the training set and helps control overfitting [30]. We use them in all kernel models for the sake of fair comparison with prior models cited [2], [3], [5]. Our implementation of bilinear models [5], [20] cannot handle full kernels save for very small sample sizes. Neither the bilinear nor bundle methods are scalable in the kernel size. The presence of high correlations between kernel matrix columns, which could result in overfitting, motivates the use of reduced kernels as much as reducing computational complexity. We do not study the effect of kernel reduction on accuracy, and follow the lead of [2], [3], [5].

5 Nonsmooth Nonconvex Bundle Method

This section presents a state-of-the-art nonsmooth nonconvex bundle method algorithm [23].

Bundle methods minimize nonsmooth locally Lipschitz convex functions:

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}). \tag{18}$$

The key idea involves approximating, at each iteration, the convex function using the maximum of a set of affine functions [31], [32], [33]. The affine functions are linearizations of the function at the points generated at the previous iterations. For the convex case, these affine functions support the function from below, and adding more of them improves the approximation of the objective.

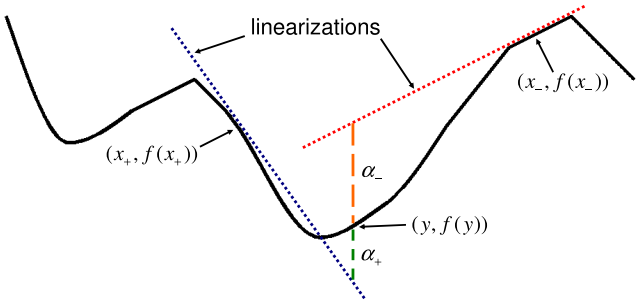


Fig. 3. Affine approximation of nonsmooth nonconvex function f at some points $\boldsymbol{x}$ either under or overestimates f at stability center $\boldsymbol{y}$. The linearizations are shown as dotted lines and the linearization errors $\alpha_+ > 0$ or $\alpha_- < 0$ appear as dashed lines.

For nonconvex objectives, such as the ones presented in the previous section, the affine functions in the bundle algorithm do not necessarily support the function. They provide local lower or upper bounds to the function valid at a certain point, say $\boldsymbol{y}$, known as the stability center and coinciding with the best current point in terms of objective function value. Fig. 3 illustrates how the bundle method deals with nonconvexity. In particular, at the current iteration, it maintains lower and upper polyhedral approximations

$$\hat{f}_+(\boldsymbol{x}) = \max_{i \in I_+} \quad f(\boldsymbol{x}_i) + \boldsymbol{g}_i^T(\boldsymbol{x} - \boldsymbol{x}_i), \tag{19}$$

and

$$\hat{f}_-(\boldsymbol{x}) = \min_{i \in I_-} \quad f(\boldsymbol{x}_i) + \boldsymbol{g}_i^T(\boldsymbol{x} - \boldsymbol{x}_i), \tag{20}$$

where $\boldsymbol{x}_i$ are the points generated at the previous iterations and $\boldsymbol{g}_i \in \partial f(\boldsymbol{x}_i)$. The points $\boldsymbol{x}_i$ are classified into two sets according to the rules $I_+ = \{i | \alpha_i \geq 0\}$ and $I_- = \{i | \alpha_i < 0\}$ based on linearization errors α_i at stability center $\boldsymbol{y}$ (which is the current solution):

$$\alpha_i = f(\boldsymbol{y}) - [f(\boldsymbol{x}_i) + \boldsymbol{g}_i^T(\boldsymbol{y} - \boldsymbol{x}_i)]. \tag{21}$$

These approximations are locally valid about this stability center. Hence, at each iteration, a new point $\boldsymbol{x}_{i+1}$ tradeoffs minimizing $\hat{f}_+(\boldsymbol{x})$ and proximity $\boldsymbol{d} = \boldsymbol{x} - \boldsymbol{y}$ under constraint $\hat{f}_+(\boldsymbol{x}) \leq \hat{f}_-(\boldsymbol{x})$. This can be converted to a direction finding problem called the *bundle method subproblem* QP with proximity control parameter $\gamma > 0$:

$$\begin{array}{rcllll} (v^\star, \boldsymbol{d}^\star) & = & \arg\min & \gamma v & + & \frac{1}{2}\boldsymbol{d}^T\boldsymbol{d} \\ & & \text{s.t.} & v & \geq & \boldsymbol{g}_i^T\boldsymbol{d} - \alpha_i, \quad i \in I_+ \\ & & & v & \leq & \boldsymbol{g}_i^T\boldsymbol{d} - \alpha_i, \quad i \in I_-. \end{array} \tag{22}$$

Scalar $v \leq 0$ is the expected improvement in f, based on the current approximation. Proximity control is essential as cuts may not support the function. The bundle method as implemented by us is stated as Algorithm 1. It calls subroutine `LinearizationError`, appearing as Algorithm 2. In practice, we solve the dual of problem (22) as did Fuduli et al. [23].

Algorithm 1. Nonconvex nonsmooth bundle method. Solves a nonconvex nonsmooth unconstrained optimization problem.

Select starting stability center $\boldsymbol{y}$ and starting proximity control parameter γ_1.
Select descent coefficient $m \in (0,1)$, tolerances δ and ϵ, and decay coefficient ω.
Initialize counter $i = 1$, starting point $\boldsymbol{x}_1 = \boldsymbol{y}$, and empty sets I_+, I_-, and J_+.
Compute function value $f(\boldsymbol{x}_1)$ and subgradient $\boldsymbol{g}(\boldsymbol{x}_1) \in \partial f(\boldsymbol{x}_1)$.
Execute `LinearizationError` (Algorithm 2) for $k = 1$.
repeat
  Solve the dual of bundle method subproblem QP (22) and determine the associated primal variables.
  Determine new point $\boldsymbol{x}_{i+1} = \boldsymbol{y} + \boldsymbol{d}^\star$.
  Compute $f(\boldsymbol{x}_{i+1})$ and $\boldsymbol{g}(\boldsymbol{x}_{i+1}) \in \partial f(\boldsymbol{x}_{i+1})$.
  Execute `LinearizationError` for $k = i + 1$.
  if $f(\boldsymbol{x}_{i+1}) \leq f(\boldsymbol{y}) + mv^\star$ **then**
    Update stability center $\boldsymbol{y} = \boldsymbol{x}_{i+1}$. Clear index sets I_+, I_- and J_+.
    for $j = 1 \ldots i + 1$ **do**
      Execute `LinearizationError` for $k = j$ to update values for the new stability center.
    end for
    if inequality (23) is satisfied **then**
      Exit.
    end if
    Solve subgradient convex combination QP (25).
    Compute $\boldsymbol{g}^\star$ (24).
    if inequality (26) is satisfied **then**
      Exit.
    end if
  else
    Decrease proximity control $\gamma \leftarrow \omega\gamma$.
  end if
  Increment i by 1.
until algorithm exits.
Return current stability center $\boldsymbol{y}$ as the solution.

Algorithm 2. LinearizationError. Computes the linearization error α at point k and associated properties.

Evaluate α_k (21).
Evaluate $a_k = \frac{\|\boldsymbol{y} - \boldsymbol{x}_k\|_2}{\|\boldsymbol{y}\|_2}$.
if $\alpha_k \geq 0$ **then**
  Add k to index set I_+.
  if $a_k \leq \epsilon$ **then**
    Add k to index set J_+.
  end if
else
  Add k to index set I_-.
end if

The bundle method exits when the subgradient associated with a new stability center is *small*. This occurs in one of two ways. First, if inequality

$$\frac{\boldsymbol{g}_{rms}(\boldsymbol{y})}{f(\boldsymbol{y})} \leq \delta \tag{23}$$

is satisfied, then local optimality is attained within algorithm tolerance δ. The vector RMS was defined earlier in Section 3. The second criterion considers that a subgradient may not be unique. We therefore estimate the minimum subgradient in a neighborhood of $\boldsymbol{y}$. To do this, we form matrix G, whose rows are the subgradients $\boldsymbol{g}_j$ have indices j in set J^+ (see Algorithm 2). This estimated minimum subgradient is computed as

$$\boldsymbol{g}^\star = G\boldsymbol{\lambda}^\star, \tag{24}$$

with convex combination weights $\boldsymbol{\lambda}^\star$ found as the solution of QP

$$\begin{array}{rcll} \boldsymbol{\lambda}^\star & = & \arg\min & \boldsymbol{\lambda}^T G^T G \boldsymbol{\lambda} \\ & & \text{s.t.} & \boldsymbol{e}^T \boldsymbol{\lambda} = 1 \\ & & & \boldsymbol{\lambda} \geq 0. \end{array} \tag{25}$$

If

$$\frac{\boldsymbol{g}^\star_{rms}(\boldsymbol{y})}{f(\boldsymbol{y})} \leq \delta, \tag{26}$$

we consider that a local minimum has been found.

Detailed information on this algorithm appears in Fuduli et al. [23].

6 Data Sets

This section describes the comprehensive database of novel and existing data sets used in this study. Descriptive statistics for all data sets appear in Table 1.

6.1 Metabolic Liability Data Sets

The goal of quickly calculating models from metabolic liability data sets, generated at the Rensselaer Exploratory Center for Cheminformatics Research (RECCR, http://reccr. chem.rpi.edu/), motivates this paper. Models foster better understanding of metabolic liability as part of the drug discovery process. The MIRank task is to predict, for each compound (box), a potential site of metabolic liability (bag) that is an experimental site. Features describe the topology and quantum chemistry of atoms, including charge, reactivity, nucleophilicity, energy, surface area, span, and bond orders.

The original $\mathrm{CYP3A4}_{2008}$ data set [20] is limited to hydrogen atoms. We present an expanded $\mathrm{CYP3A4}_{2009}$ data set with an increased number of boxes and features, and including all compounds reported by Sheridan et al. [34]. This 3A4 data set also describes all compound atoms. Additionally, this paper introduces eight new CYP data sets for MIRank modeling. See Table 1. Details on molecular structures and laboratory experiments are found elsewhere [35], [36].

Moreover, all CYP data sets are folded together to obtain one large data set that we call CYP_{2009}. Modeling on this comprehensive data set assumes that the chemical mechanisms underlying degradation are the same across all isozymes, an assumption that is only partially correct, but which permits modeling on a larger data set having a sample size expected to be available within 2 years.

TABLE 1
Data Set Properties for 19 MIRank and 6 MIC Problems, Respectively

dataset	number of features	number of boxes	number of bags	number of items
MIRank				
$CYP1A2_{2009}$	540	55 compounds	890 sites	1946 atoms
$CYP2A6_{2009}$	540	30 compounds	429 sites	975 atoms
$CYP2B6_{2009}$	540	44 compounds	728 sites	1681 atoms
$CYP2C8_{2009}$	540	28 compounds	511 sites	1155 atoms
$CYP2C9_{2009}$	540	133 compounds	2497 sites	5268 atoms
$CYP2C19_{2009}$	540	43 compounds	715 sites	1613 atoms
$CYP2D6_{2009}$	540	156 compounds	2827 sites	6516 atoms
$CYP2E1_{2009}$	540	37 compounds	412 sites	964 atoms
$CYP3A4_{2008}$	36	227 compounds	2260 sites	4823 atoms
$CYP3A4_{2009}$	540	397 compounds	8245 sites	19614 atoms
CYP_{2009}	540	923 compounds	17236 sites	39732 atoms
census16h	16	50 states	376 divisions	3054 towns
census16l	16	50 states	376 divisions	3054 towns
census32	32	50 states	376 divisions	3054 towns
MIC				
musk1	166	–	92 compounds	476 conformations
musk2	166	–	102 compounds	6581 conformations
elephant	230	–	200 images	1391 segments
fox	230	–	200 images	1320 segments
tiger	230	–	200 images	1220 segments

6.2 United States Census Data Sets

The census-16h and census-16l data sets belong to the Data for Evaluating Learning in Valid Experiments (DELVE, http://www.cs.toronto.edu/~delve/) repository. They have been previously adapted to fit the MIRank framework [20]. The task is to predict, for each state (box), the division (bag) that contains the town (item) with the highest median housing unit price. We combine the features from census-16h and census-16l to form combined data set census-32.

6.3 Musk Data Sets

The original MIC data sets were the musk1 and musk2 data sets [1]. These are available in the UCI repository (http://archive.ics.uci.edu/ml/). The task is to predict whether a compound is a musk or not. Bags correspond to compounds, and each bag consists of a number of molecular conformations that are the items.

6.4 Animal Image Annotation Data Sets

Three MIC image annotation data sets (elephant, fox, tiger) [2] are available at http://www.cs.columbia.edu/~andrews/. The task involves classifying images (bags) based on their segments (items). An animal appears in an image (bag) if at least one segment (item) therein contains this animal.

7 Implementation

The implementation of the loss functions and bundle method presented in Sections 4-5 is called $\mathrm{MIL}^{\mathrm{bundle}}$. Both classification ($\mathrm{MIC}^{\mathrm{bundle}}$) and ranking ($\mathrm{MIRank}^{\mathrm{bundle}}$) losses are available. This was implemented in Matlab version 7 (R14) (http://www.mathworks.com/). Adrian Wills' `qpas` (QPC project, http://sigpromu.org/quadprog/), a dual active-set implementation written in C and compiled in Matlab, solves the bundle QP subproblems. The bundle method sets descent coefficient $m = 0.1$, tolerances $\delta = 10^{-2}, \epsilon = 10^{-1}$, decay coefficient $\varsigma = 0.9$, and starting proximity control parameter $\gamma_1 = 1$.

These bundle algorithms are compared to the bilinear algorithms $\mathrm{MIC}^{\mathrm{bilinear}}$ and $\mathrm{MIRank}^{\mathrm{bilinear}}$. The former is our implementation of MICA [5]. The latter is our own [20], available at http://reccr.chem.rpi.edu/MIRank/. As in Mangasarian and Wild [5], bilinear algorithms exit after a maximum of 80 iterations.

Data are centered and scaled using the mean and standard deviation across all items, and then thresholded so that all features fall within range $[-5, 5]$. In a small number of cases in the CYP2E1 data set, infinite or missing feature values were set to 0. Additionally, for one 2E1 compound, no preferred site of metabolism is provided, and so this compound is dropped.

We performed 10-fold cross-validation (CV), except for MIRank data sets with more than 200 boxes, in which case we performed 5-fold CV. We describe k-fold CV. Boxes (for MIRank) or bags (for MIC) were randomly split into k equal partitions so that, at each split, one partition is assigned to testing, one partition goes to validation, and the remaining partitions are used for training. For MIC, these partitions are stratified by bag label. Nine logarithmically spaced values for tradeoff ν over range $[10^{-4}, 10^{4}]$ with one value per decade are considered. Models for this 1D grid are calculated using the training set and assessed using the validation set, and the performance of this best model on the test set is recorded. This is repeated k times so that each partition is used once in validation and once in testing.

This entire process is repeated 10 times, reshuffling samples into 10 partitions each time. For each repeat, results are averaged, and then the mean and standard deviation across the repeats are reported in Tables 2 and 3.

For nonlinear learning, a few differences appear. The Gaussian radial basis function (RBF)

$$k(\boldsymbol{x}_m, \boldsymbol{x}_n) = \exp\left(-\frac{\|\boldsymbol{x}_m - \boldsymbol{x}_n\|_2^2}{\tau^2}\right), \quad \tau > 0, \tag{27}$$

is chosen. Thirteen logarithmically spaced values for RBF width τ over range $[2^{-6}, 2^{6}]$ are considered, resulting in a

TABLE 2
Multiple-Instance Ranking Results

dataset kernel	random expectation	$\text{MIRank}^{\text{bilinear}}$ [20] linear	$\text{MIRank}^{\text{bilinear}}$ linear	$\text{MIRank}^{\text{bundle}}$ linear	$\text{MIRank}^{\text{bundle}}$ Gaussian
CYP1A2_{2009}	24.1	−−	61.1 ± 6.3	70.5 ± 4.0	63.3 ± 3.7
CYP2A6_{2009}	23.0	−−	44.0 ± 6.9	57.3 ± 4.0	46.6 ± 6.6
CYP2B6_{2009}	22.4	−−	53.7 ± 7.6	67.7 ± 4.2	59.5 ± 4.3
CYP2C8_{2009}	22.6	−−	52.1 ± 9.2	57.6 ± 4.9	51.3 ± 8.2
CYP2C9_{2009}	17.5	−−	58.4 ± 3.9	70.0 ± 3.5	56.0 ± 1.8
CYP2C19_{2009}	19.2	−−	53.1 ± 7.6	62.7 ± 3.3	48.8 ± 5.6
CYP2D6_{2009}	16.5	−−	61.8 ± 3.2	72.0 ± 2.4	66.7 ± 1.7
CYP2E1_{2009}	35.6	−−	59.8 ± 4.6	66.0 ± 6.5	62.1 ± 6.9
CYP3A4_{2008}	31.4	70.9	69.3 ± 1.5	70.8 ± 1.6	68.6 ± 1.6
CYP3A4_{2009}	18.6	−−	54.7 ± 5.3	75.2 ± 1.9	53.0 ± 1.1
CYP_{2009}	19.6	−−	65.4 ± 0.1	77.4 ± 1.4	60.7 ± 0.6
census16h	29.6	60.3	58.8 ± 5.2	63.0 ± 4.4	62.6 ± 5.0
census16l	29.6	57.5	56.4 ± 7.5	66.0 ± 2.3	60.0 ± 4.8
census32	29.6	−−	62.6 ± 4.1	65.0 ± 3.1	60.2 ± 4.9

The ranking accuracy is the percentage of boxes for which a preferred bag is correctly identified, permitting two guesses per box.

cross-validation over a 2D grid to set ν and τ. A reduced kernel matrix strategy (see Section 4) with 500 columns is used (save when the full kernel matrix has fewer columns). Columns are stratified by box or bag to ensure that large boxes or bags do not dominate the reduced kernel.

8 Ranking Accuracy for MIRank Data Sets

Table 2 reports results for MIRank modeling using our bilinear algorithm [20] and this paper's bundle algorithm. The ranking accuracy is the percentage of boxes for which a preferred bag is correctly identified, permitting two guesses per box [20], [34], [37]. This table includes results previously reported for three data sets using $\text{MIRank}^{\text{bilinear}}$. The expected accuracy of a model that randomly ranks bags for each box is also reported.

As seen from columns 3 and 4 of Table 2, the results of this paper are consistent with those previously published using CYP3A4_{2008}, allowing for some variation due to experimental design differences. This is expected, as we used the same code to generate these replicate results.

The goal behind this table is to find modeling accuracies using the bundle algorithm that are at-par or better than those obtained using the bilinear algorithm, thereby justifying the use of a faster method. Considering the CYP data sets, the improvement of $\text{MIRank}^{\text{bundle}}$ over $\text{MIRank}^{\text{bilinear}}$ averages 20 percent. None of the data sets fared worse with $\text{MIRank}^{\text{bundle}}$ than with $\text{MIRank}^{\text{bilinear}}$. We note that the data sets that showed bilinear results competitive with the bundle were those appearing in Bergeron et al. [20] (CYP3A4_{2008} and census). These data sets have the common trait of having a small number of features. These results suggest that the bundle method is more resistant to overfitting than the bilinear algorithm. Several explanations for this gap exist. For instance, both approaches are susceptible to finding local minima that are not global. However, the bilinear formulation may be more susceptible to overfitting as each preferred bag possesses a vector of convex combination weights treated as variables that handle the ambiguity of which item in this bag determines the preference. This results in a high variables-to-samples ratio.

TABLE 3
Multiple-Instance Classification Results

dataset	mi-SVM [2]	MI-SVM [2]	MICA [5]	CH-FF [3]	$\text{MIC}^{\text{bilinear}}$	$\text{MIC}^{\text{bundle}}$
linear						
musk1	−−	−−	−−	−−	73.8 ± 2.2	75.6 ± 3.2
musk2	−−	−−	−−	−−	72.1 ± 2.9	76.8 ± 1.9
elephant	82.2	81.4	80.5	82.4	76.3 ± 2.6	80.5 ± 1.7
fox	58.2	57.8	58.7	60.8	55.9 ± 1.6	58.3 ± 2.8
tiger	78.4	84.0	82.6	82.2	74.1 ± 2.8	79.1 ± 1.8
Gaussian						
musk1	87.4	77.9	84.4	88.8	82.1 ± 4.8	84.1 ± 1.4
musk2	83.6	84.3	90.5	85.7	78.4 ± 0.0	85.2 ± 3

The classification accuracy is the percentage of correctly classified bags. Note that several papers do not report linear modeling results for the Musk data sets.

The metabolic liability models reported here for CYPs 2C9, 2D6, and 3A4 are among the best reported in the literature to date. Sheridan et al. [34] reported random forest models with accuracy 68-73 percent for CYP2C9, 70-72 percent for 2D6, and 74-77 percent for 3A4. Comparing with our results in Table 2, we match their performances. Direct comparisons with [34] are difficult since their experimental design is unclear and their features are private. We first present models for ranking sites of metabolic liability in drug leads by CYPs 1A2, 2A6, 2B6, 2C8, 2C19, and 2E1. Each model is calculated from a small database of compounds, and we fully expect our algorithms to produce improved predictions as more data for these isozymes is accessed. This expectation is based on our observations on the CYP3A4_{2009} data set. Fig. 4 displays the relation between training set size and ranking accuracy for this data set, and was generated by varying the training set sizes. Validation and testing set sizes remained fixed as they would be for 5-fold CV. We notice that prediction accuracies level out beyond a training set size of approximately 150 molecules.

We note from Table 2 that the standard deviations across repeated cross-validations decrease with sample size, as would be expected. This confirms that, with the aim of generating models quickly in a pharmaceutical enterprise setting, a single cross-validation is sufficient when enough compounds form a CYP database.

Models generated in this paper are predictive so as to make metabolism predictions on new molecules. For example, we used our linear CYP3A4_{2009} model to accurately predict the site of experimental metabolism in 16 out of 20 (or 80 percent) proprietary compounds whose structures were provided by a major pharmaceutical company. Models are

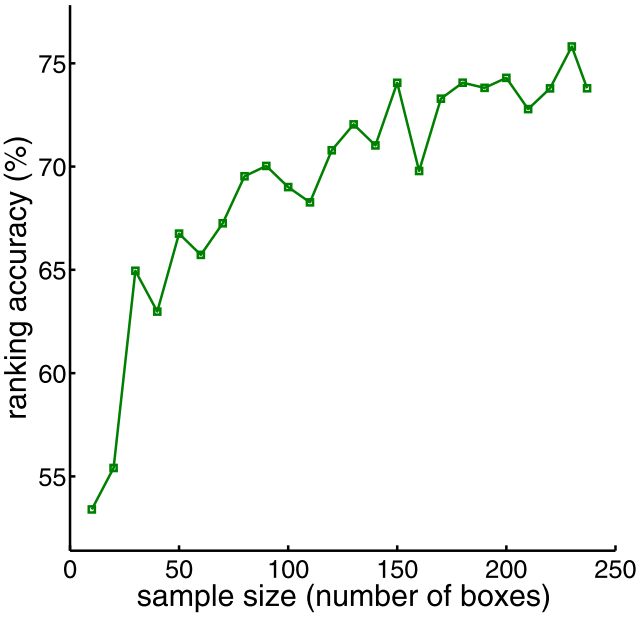


Fig. 4. Ranking accuracy increases with sample size. Cross-validation was performed on $\mathrm{CYP3A4}_{2009}$, for various training set sizes.

also explicative so as to learn properties within a data set. For instance, we determined that the three highest-weighted features in this model are:

1. `NA_0_S`: a binary feature that identifies the atom as a sulfur or not;
2. `atom_area`: the solvent accessible surface area;
3. `Span`: a topological measure of distance from the atom to the molecule center.

Such lists may be used by chemists to understand and address problematic sites of metabolic liability on drug-like molecules. A detailed analysis of this process is being prepared for a medicinal chemistry audience.

Features believed to be relevant for this application continue to be developed. Between the $\mathrm{CYP3A4}_{2008}$ and $\mathrm{CYP3A4}_{2009}$ data sets, the number of compounds and features increased by 66 percent and 14 times, respectively, with little change in performance. However, determining the impact of new features will be made easier with the fast $\mathrm{MIRank}^{\mathrm{bundle}}$.

One strong motivation toward the goal of implementing a nonsmooth nonconvex bundle method to build MIL models was our desire to boost predictive performances with kernels. Surprisingly, as seen in Table 2, linear models performed better on all data sets. We find that CYP kernel models obtained worse solutions than their linear counterparts, but fared better on those data sets which have a smaller features-to-samples ratio. Other MIL studies report the strength of linear models over kernel ones—considering the MIC models in Andrews et al. [2], the linear outperforms Gaussian in 9 out of 10 cases. Clearly, a study into suitable kernel functions for MIL is needed. Such investigation will be aided by the availability of fast $\mathrm{MIRank}^{\mathrm{bundle}}$.

9 Classification Accuracy for MIC Data Sets

Table 3 reports results for MIC modeling. The accuracy is the proportion of bags that are correctly classified, expressed as a percentage. This table includes results previously reported for four different SVM-like models (mi-SVM and MI-SVM [2], MICA [5] and CH-FF [3]) as well as those implemented for this paper. For all tasks, $\mathrm{MIC}^{\mathrm{bundle}}$ finds models that are within the range of previously published ones.

As was the case with MIRank, the goal behind Table 3 is to find modeling accuracies using the bundle algorithm that are at least at-par with previous results, so as to support the use of our faster methods. This table finds $\mathrm{MIC}^{\mathrm{bundle}}$ models that are in the range of previously reported ones.

Results reported using MICA and $\mathrm{MIC}^{\mathrm{bilinear}}$ (which is our implementation of MICA) should be similar. Yet, we observe a 5 to 8 point drop between these. This exemplifies the difficulty of reimplementing an algorithm and the differences in performance that are induced by such things as the experimental design, data processing, choice of solver, and solver parameters. Such differences may extend to the results with $\mathrm{MIC}^{\mathrm{bundle}}$.

10 Execution Speed

The computational cost associated with learning MIL models is investigated. A theoretical assessment of the computational effort is difficult to determine for both algorithms, as the number of subproblems that are solved is not known. However, it is known that the number of variables, inequality constraints, and box constraints in the LPs that constitute the bilinear subproblems all scale with the smaller of sample size ℓ or number n of features (assuming the better of the primal or dual solution strategy is selected), and that the bundle subproblem QP scales with the smaller of n or bundle size.

The computational cost associated with learning MIL models is empirically investigated on the comprehensive CYP_{2009} data set. The tradeoff hyperparameter is set to $\nu = 1$. Models are learned using $\mathrm{MIRank}^{\mathrm{bilinear}}$ and $\mathrm{MIRank}^{\mathrm{bundle}}$ on the same computer from an increasingly large number of boxes. This is repeated 10 times on random shuffles of the boxes.

For each training set size, the median execution time to train a model is plotted to produce Fig. 5. Note that the bundle algorithm uses a small gradient criterion, whereas the bilinear one is based on function value change. To produce these scalability plots, both algorithms exit when a set accuracy is reached. For MIRank, this stopping ranking accuracy is 66.6 percent. For MIC, the stopping classification accuracy is set lower, at 50 percent, as some smaller musk training sets cannot reach a higher threshold.

We find that $\mathrm{MIRank}^{\mathrm{bundle}}$ is empirically linearly scalable. This means that doubling the number of samples doubles the execution time. At the same time, we find that $\mathrm{MIRank}^{\mathrm{bilinear}}$ blows up as sample size increases. This is due to the fact that the number of variables in bundle method subproblem QP (22) does not grow with training set size, while the number of constraints in the bilinear algorithm's subproblem LPs does [20]. Further, the number of constraints in the bundle QP can be kept reasonable by an appropriate aggregation strategy [23].

Convergence of the nonsmooth nonconvex bundle algorithm to a local solution has been theoretically established [23], but its convergence rate remains an open question. Even bundle algorithms for convex losses do not prove scalability [21], [22]; their use to solve support vector

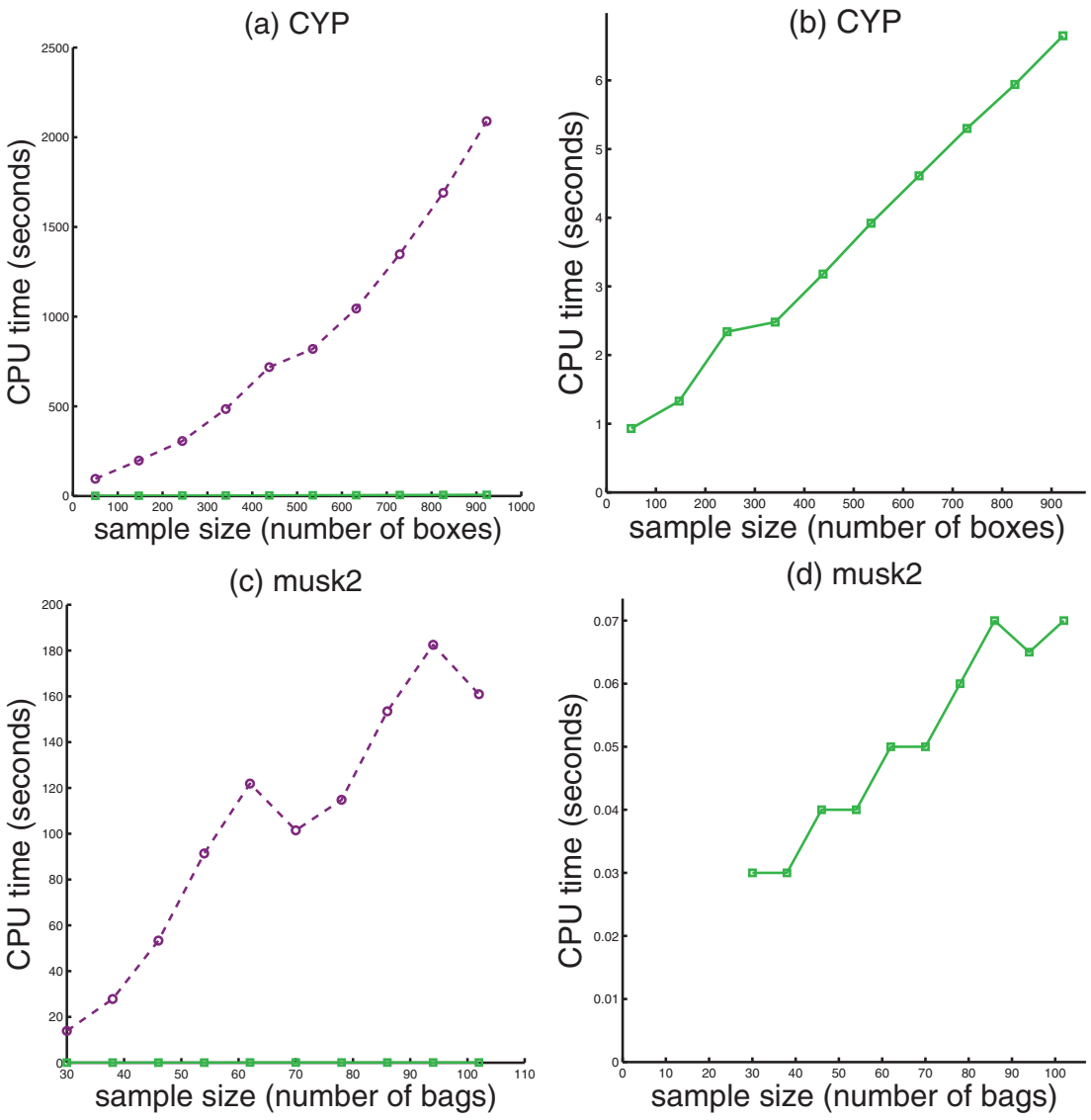


Fig. 5. Model training times. The dashed purple curve in (a), (c) denotes $\mathrm{MIRank}^{\mathrm{bilinear}}$ model training time in CPU seconds with increasing sample size (number of boxes or bags, as the case may be). The full green curve straddling the horizontal axis denotes the same for $\mathrm{MIRank}^{\mathrm{bundle}}$. Empirical linear scalability for $\mathrm{MIRank}^{\mathrm{bundle}}$ is better exhibited in (b), (d). Sample size is measured in number of bags. Clearly, the empirical speed for $\mathrm{MIRank}^{\mathrm{bundle}}$ is much less than for $\mathrm{MIRank}^{\mathrm{bilinear}}$.

machines is based on empirical evidence of their speed. However, we can make some qualitative comparisons between the bundle and bilinear solvers. Consider a data set consisting of ℓ samples and n features.

The bundle algorithm minimizes an unconstrained loss possessing $n+1$ variables through solving a sequence of subproblem QPs (22) in $n+1$ variables and p inequality constraints, where p is the bundle size at a given point. These small subproblem QPs are solved in polynomial time. On the flipside, the bilinear formulation consists of $2\ell+2n$ samples, $\ell+2n$ inequality constraints, and $2\ell+n$ box constraints [20]. The algorithm alternates between two LP subproblems; the larger one is comprised of 2ℓ samples, ℓ inequality constraints, and 2ℓ box constraints. These small subproblem LPs are solved in polynomial time. For each algorithm, the unknown factor is how many subproblems are solved before exit criteria are satisfied. Despite this, we can appreciate that the bundle speed stems from solving very small subproblems.

We suspect that linear scalability will not be observed for all nonconvex objectives, but we speculate that it can be proven for the MIC and MIRank since it has been observed in all experiments performed to date. Even without theoretical guarantees, the observed reduction in computational effort by switching from the bilinear to bundle algorithms constitutes a necessary requirement for a proposed online metabolic liability prediction tool based on MIRank models.

11 Conclusion

This paper proposes the first linearly scalable algorithm to solve multiple-instance learning problems using a nonsmooth nonconvex bundle algorithm. While its computational complexity has not been formally established,

$\text{MIL}^{\text{bundle}}$ exhibits linear time scalability in the sample size. This allows for new and larger data sets to be analyzed. Thereupon, MIL problems may be studied in greater detail, allowing for models to be built from larger sample sizes, more features, and permitting more complex tasks, including feature selection. We hope that freely distributing the data sets and codes used in this paper (http://www.rpi.edu/~bennek/MIRank) will support further research in MIL, as well as assist other researchers in exploiting these powerful new nonconvex bundle methods on other learning problems.

Results on MIC and MIRank illustrate that the bundle method can be readily adapted to different loss functions. We chose the least-squares penalty for the empirical and structural risk terms to obtain more informative subgradients as a first start. In principle, the method could readily optimize the 1-norm structural and empirical risks used in prior bilinear MIL models [5], [20]. A future paper will investigate the impact of model selection decisions such as these.

Comparisons with the MIRank algorithm of Hu et al. [19] were not made for two reasons. First, their best model was based on a loss containing the softmax operator that we have not considered. With the speed afforded by the nonconvex bundle algorithm, we foresee a future comparison of various losses for MIRank, including appropriate penalty of empirical risk and structural risk terms. Second, they solve successive QPs, where the size of each QP is not linearly bounded by sample size, resulting in an approach that does not scale linearly with sample size, as with the successive LP method.

We find that the comprehensive CYP_{2009} data set has better predictive ability across all compounds than any of the individual CYP models. This is an interesting result: A global model achieves improved predictions over isozyme-specific ones. This speaks to the similarities in metabolism mechanisms across isozymes, but may also simply reflect a common reliance upon reactivity that is better represented in this larger data set. We did not exploit compound overlap in CYP data sets; incorporating multitask learning into MIRank may further improve results. Attempts at multitask MIL would be facilitated by the fast bundle method and could easily be attempted by changing the loss function.

Acknowledgments

This work was funded by the *Fonds québécois de la recherche sur la nature et les technologies* doctoral fellowship program and grants by the US Office of Naval Research (number N00014-06-1-0014) and the US National Institutes of Health (number R01LM009731). The authors thank Michael Krein and Tao-wei Huang for support with computational issues and data set generation, Dawnmarie Robens for her expert administrative support and caring resourcefulness, and the group of people at the Rensselaer Exploratory Center for Cheminformatics Research (RECCR).

References

[1] T.G. Dietterich, R.H. Lathrop, and T. Lozano-Perez, "Solving the Multiple Instance Problem with Axis-Parallel Rectangles," *Artificial Intelligence*, vol. 89, nos. 1/2, pp. 31-71, 1997.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support Vector Machines for Multiple-Instance Learning," *Proc. Advances in Neural Information Processing Systems*, vol. 15, 2003.

[3] M.M. Dundar, G. Fung, B. Krishnapuram, and R.B. Rao, "Multiple-Instance Learning Algorithms for Computer-Aided Detection," *IEEE Trans. Biomedical Eng.*, vol. 55, no. 3, pp. 1015-1021, Mar. 2008.

[4] J.F. Murray, G.F. Hughes, and K. Kreutz-Delgado, "Machine Learning Methods for Predicting Failures in Hard Drives: A Multiple-Instance Application," *J. Machine Learning Research*, vol. 6, pp. 783-816, 2005.

[5] O.L. Mangasarian and E.W. Wild, "Multiple Instance Classification via Successive Linear Programming," *J. Optimization Theory and Applications*, vol. 137, no. 3, pp. 555-568, 2008.

[6] J. Wang and J. Zucker, "Solving the Multiple-Instance Problem: A Lazy Learning Approach," *Proc. Int'l Conf. Machine Learning*, vol. 17, pp. 1119-1125, 2000.

[7] T. Gärtner, P. Flach, A. Kowalczyk, and A. Smola, "Multi-Instance Kernels," *Proc. 19th Int'l Conf. Machine Learning*, vol. 19, pp. 179-186, 2002.

[8] N. Weidmann, E. Frank, and B. Pfahringer, "A Two-Level Learning Method for Generalized Multi-Instance Problems," *Proc. European Conf. Machine Learning*, pp. 468-479, 2003.

[9] P. Auer and R. Ortner, "A Boosting Approach to Multiple Instance Learning," *Proc. European Conf. Machine Learning*, vol. 15, pp. 63-74, 2004.

[10] Y. Chen and J. Wang, "Image Categorization by Learning and Reasoning with Regions," *J. Machine Learning Research*, vol. 5, pp. 913-939, 2004.

[11] H. Blockeel, D. Page, and A. Srinivasan, "Multi-Instance Tree Learning," *Proc. 22nd Int'l Conf. Machine Learning*, vol. 22, pp. 144-152, 2005.

[12] Q. Tao, S. Scott, N.V. Vinodchandran, T. Osugi, and B. Mueller, "Kernels for Generalized Multiple-Instance Learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 12, pp. 2084-2097, Dec. 2008.

[13] O. Maron and A.L. Ratan, "Multiple-Instance Learning for Natural Scene Classification," *Proc. 15th Int'l Conf. Machine Learning*, vol. 15, 1998.

[14] Q. Zhang and S.A. Goldman, "EM-DD: An Improved Multiple-Instance Learning Technique," *Advances in Neural Information Processing Systems*, vol. 14, pp. 1073-1080, 2001.

[15] J. Ramon and L.D. Raedt, "Multi Instance Neural Networks," *Proc. 17th Int'l Machine Learning Conf.*, vol. 17, 2000.

[16] S. Ray and D. Page, "Multiple Instance Regression," *Proc. Int'l Conf. Machine Learning*, vol. 18, pp. 425-432, 2001.

[17] D. Dooly, Q. Zhang, S. Goldman, and R. Amar, "Multiple-Instance Learning of Real-Valued Data," *J. Machine Learning Research*, vol. 3, pp. 651-678, 2002.

[18] O. Wu, J. Gao, W. Hu, B. Li, and M. Zhu, "Indentifying Multi-Instance Outliers," *Proc. SIAM Int'l Conf. Data Mining*, vol. 14, pp. 430-441, 2010.

[19] Y. Hu, M. Li, and N. Yu, "Multiple Instance Ranking: Learning to Rank Images for Image Retrieval," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[20] C. Bergeron, J. Zaretzki, C. Breneman, and K.P. Bennett, "Multiple Instance Ranking," *Proc. 25th Int'l Conf. Machine Learning*, pp. 48-55, 2008.

[21] T. Joachims, "Training Linear SVMs in Linear Time," *Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 217-226, 2006.

[22] C.H. Teo, Q.V. Le, A. Smola, and S.V.N. Vishwanathan, "A Scalable Modular Convex Solver for Regularized Risk Minimization," *Proc. 13th ACM Conf. Knowledge Discovery and Data Mining*, pp. 727-736, 2007.

[23] A. Fuduli, M. Gaudioso, and G. Giallombardo, "Minimizing Nonconvex Nonsmooth Functions via Cutting Planes and Proximity Control," *SIAM J. Optimization*, vol. 14, no. 3, pp. 743-756, 2004.

[24] A. Astorino and A. Fuduli, "Nonsmooth Optimization Techniques for Semisupervised Classification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2135-2142, Dec. 2007.

[25] F.P. Guengerich, "Cytochrome p450 and Chemical Toxicology," *Chemical Research in Toxicology*, vol. 21, no. 1, pp. 70-83, 2008.

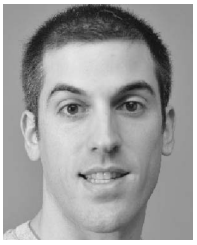
[26] F.H. Clarke, *Optimization and Nonsmooth Analysis*. Wiley, 1983.

[27] A. Ruszczyński, *Nonlinear Optimization*. Princeton Univ. Press, 2006.

[28] N.J. Higham, *Accuracy and Stability of Numerical Algorithms*, second ed. SIAM Press, 2002.

[29] B.E. Boser, I.M. Guyon, and V.N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proc. Fifth Ann. ACM Workshop Computational Learning Theory*, pp. 144-152, 1992.
[30] Y.-J. Lee and O.L. Mangasarian, "RSVM: Reduced Support Vector Machines," *Proc. SIAM Int'l Conf. Data Mining*, 2001.
[31] C. Lemaréchal, "An Algorithm for Minimizing Convex Functions," *Proc. Int'l Federation for Information Processing Congress*, pp. 552-556, 1974.
[32] P. Wolfe, "A Method of Conjugate Subgradients for Minimizing Non-Differentiable Functions," *Nondifferentiable Optimization*, M. Balinski and P. Wolfe, eds., pp. 145-173, Springer, 1975.
[33] M.M. Makela, "Survey of Bundle Methods for Nonsmooth Optimization," *Optimization Methods and Software*, vol. 17, no. 1, pp. 1-29, 2001.
[34] R.P. Sheridan, K.R. Korzekwa, R.A. Torres, and M.J. Walker, "Empirical Regioselectivity Models for Human Cytochromes P450 3A4, 2D6, and 2C9," *J. Medicinal Chemistry*, vol. 50, pp. 3173-3184, 2007.
[35] S. Rendic, "Summary of Information on Human CYP Enzymes: Human P450 Metabolism Data," *Drug Metabolism Rev.*, vol. 34, nos. 1/2, pp. 83-448, 1997.
[36] C.M. Brown, B. Reisfeld, and A.N. Mayeno, "Cytochromes P450: A Structure-Based Summary of Biotransformations Using Representative Substrates," *Drug Metabolissm Rev.*, vol. 40, pp. 1-100, 2008.
[37] S.B. Singh, L.Q. Shen, M.J. Walker, and R.P. Sheridan, "A Model for Predicting Likely Sites of CYP3A4-Mediated Metabolism on Drug-Like Molecules," *J. Medicinal Chemistry*, vol. 46, pp. 1330-1336, 2003.

Charles Bergeron received the PhD degree in mathematical sciences from Rensselaer Polytechnic Institute. He enjoys developing customized models for emerging applications, including generalizing the support vector machine to new objectives and solving the resulting optimization problems using speedy algorithms. He is a member of the IEEE and the IEEE Computer Society.



Gregory Moore received the MS and PhD degrees in mathematical sciences from Rensselaer Polytechnic Institute. He received the bachelor's degree in mathematics from Le Moyne College, Syracuse, New York. His current work includes algorithm development for the data mining model selection problem. He is a member of the IEEE.

Jed Zaretzki received the BS degree with honors in computer science from McGill University with a mathematics minor and the PhD degree in computational chemistry from Rensselaer Polytechnic Institute. His primary research focuses on descriptor development to predict sites of CYP P450 mediated metabolism for drug design.

Curt M. Breneman is acting head of the Department of Chemistry and Chemical Biology at Rensselaer Polytechnic Institute and directs the Rensselaer Exploratory Center for Cheminformatics Research. His group focuses on the development of new descriptors and techniques for modeling molecular behavior through the application of predictive cheminformatics.

Kristin P. Bennett is a professor of mathematical sciences and computer science at Rensselaer Polytechnic Institute. She is an active member of the machine learning, data mining, and operations research communities. She researches optimization approaches to modeling in applications including chemistry, biology, epidemiology, engineering, and business.