Computers & Operations Research 52 (2014) 326–333

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Multi-instance classification through spherical separation and VNS ☆

Frank Plastria [a,*], Emilio Carrizosa [b], José Gordillo [b]

[a] MOSI, Vrije Universiteit Brussel, Belgium
[b] SEIO, Universidad de Sevilla, Spain

ARTICLE INFO

Available online 17 May 2013

Keywords:
Supervised classification
Multi-instance learning
Mixed-integer programming
Variable neighborhood search

ABSTRACT

A two-class classification problem is considered where the objects to be classified are bags of instances in d-space. The classification rule is defined in terms of an open d-ball. A bag is labeled positive if it meets the ball and labeled negative otherwise. Determining the center and radius of the ball is modeled as a SVM-like margin optimization problem. Necessary optimality conditions are derived leading to a polynomial algorithm in fixed dimension. A VNS type heuristic is developed and experimentally tested. The methodology is extended to classification by several balls and to more than two classes.

1. Introduction

A Multiple Instance Problem (MI) is a supervised classification problem where bags (finite sets) of instances in $\mathbb{R}^d$ are to be labeled, +1 or −1 in the two-class case. As in usual single-instance classification there exist not only different types of classification rules but also different ways to generalize these to the MI case, see e.g. [17–20,24,25] and the references therein.

To give a flavor of some application fields, we may first mention the seminal contribution [10] where the bags were molecules and the instances different low-energy conformations. Such a molecule is observed to be positive if it may become active in producing a drug, which happens only if at least one of its conformations allows binding to a target site of a particular larger molecule. Other applications are found in image classification or retrieval where images are bags of blobs (subsets of its pixels obtained by some segmentation process) and an image is positive if it contains the representation of some given object (see e.g. [21]). In web mining one may need to suggest web pages to a user based on a request or on its browsing history; bags will then be web pages containing many links to other websites (the instances) described by their d most frequent terms and will be selected if at least one instance is close to the user's wishes.

In such applications it may be assumed that some ideal position exists representing the goal that determines that a bag is positive due to closeness by at least one of its instances, and negative otherwise. This idea naturally leads to the technique studied in this paper: separation by an open d-ball (as recently studied in e.g. [2]) adapted to MI using the so-called *presence-based MI* rule (see [25]) to label a bag as +1 if it meets the d-ball, and otherwise as −1. Training algorithms using mathematical optimization in the vein of [5] are developed to determine the ball to be used in this rule by a fitting process to a training set of pre-classified bags. The advantage of our approach over most other methods that are based on continuous optimization techniques, resides in the fact that a finite set of candidate globally optimal solutions is obtained that can be searched either by laborious full enumeration or more practically by heuristics such as VNS.

2. Formulation of the problem

Consider a database Ω containing objects $i=(X_i, Y_i)\in\Omega$, where each X_i is a finite set of feature vectors, $X_i=\{x_1, x_2, \ldots, x_{K_i}\}$, with $x_k\in\mathbb{R}^d$, $k=1,\ldots,K_i$, and where Y_i is the corresponding label +1 or −1.

The spherical classification rule we are looking for is defined in terms of an open ball with center $x_0\in\mathbb{R}^d$ and radius $r\in\mathbb{R}_+$, to be used as follows:

Given a bag $X\subset\mathbb{R}^d$,

$$\begin{aligned} &\bullet \text{ label as } +1 \quad \text{if } \exists x\in X \text{ such that } \|x-x_0\|^2-r^2<0,\\ &\bullet \text{ label as } -1 \quad \text{otherwise, i.e., if } \forall x\in X,\ \|x-x_0\|^2-r^2\geq 0. \end{aligned} \tag{1}$$

The actual rule is chosen in some optimal way with respect to a training sample $I\subset\Omega$, subdivided into the groups $G_{+1}=\{i\in I: Y_i=+1\}$ and $G_{-1}=\{i\in I: Y_i=-1\}$. Note that in the usual research environment the remaining part of the database is used for testing the quality of the methodology, while when the methodology is applied in practice the training set will equal the whole database of available classified data. In this paper we apply the maximal

☆This work was started while the second author was visiting MOSI.

* Corresponding author. Tel.: +32 2 6293609; fax: +32 2 6293690.
E-mail addresses: frank.plastria@vub.ac.be (F. Plastria), ecarrizosa@us.es (E. Carrizosa), jgordillo@us.es (J. Gordillo).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.05.009

margin approach that has been popularized in the successful Support Vector Machine (SVM) techniques (see e.g. [5,8]). However, contrary to the use of kernels usual in SVM when seeking nonlinear separation, we obtain spherical separation directly in the original feature space as in [2].

We have to maximize the margin Δ, i.e., the smallest of all slacks in (1), over all choices of an instance $x_j \in X_j$ for each bag $j \in G_{+1}$. Therefore the optimization problem we face may be stated as follows:

$$\max_{(x_j)_{j\in}\prod_{j\in G_{+1}} X_j} \max_{x_0,r,\Delta} \quad \Delta$$
$$\text{s.t.} \quad \|x_j - x_0\|^2 \le r^2 - \Delta, \ \forall j \in G_{+1}$$
$$\|x - x_0\|^2 \ge r^2 + \Delta, \ \forall x \in \bigcup_{i\in G_{-1}} X_i. \tag{2}$$

Denoting $r_{+1}^2 = r^2 - \Delta$, $r_{-1}^2 = r^2 + \Delta$, one has that $\Delta = (r_{-1}^2 - r_{+1}^2)/2$ and $r^2 = (r_{+1}^2 + r_{-1}^2)/2$, and Problem (2) can be rewritten as

$$\max_{(x_j)_{j\in}\prod_{j\in G_{+1}} X_j} \max_{x_0,r_{+1},r_{-1}} \quad r_{-1}^2 - r_{+1}^2$$
$$\text{s.t.} \quad \|x_j - x_0\|^2 \le r_{+1}^2, \ \forall j \in G_{+1}$$
$$\|x - x_0\|^2 \ge r_{-1}^2, \ \forall x \in \bigcup_{i\in G_{-1}} X_i. \tag{3}$$

Hence, an instance x_j of each bag in G_{+1} and two concentric balls $B(x_0, r_{+1})$, $B(x_0, r_{-1})$ are sought, where $B(x_0, r_{+1})$ contains all selected x_j, $B(x_0, r_{-1})$ contains no instance of any bag of G_{-1} and maximize the difference between the squared radii. We will denote a finite solution of Problem (3) by (x_0, r_{+1}, r_{-1}).

Note that if every d-ball meeting all positive bags also meets a negative bag the original problem will be unfeasible, in which case no classification rule arises. However, problem (3) is always feasible, but the latter case will have negative optimal value, corresponding to the smallest possible overlap area between the two balls, and $r^2 = (r_{+1}^2 + r_{-1}^2)/2$ then still defines a classification rule (1).

Problem (3) might also be unbounded. It can be shown that this arises if and only if MI-separation of the bags by a hyperplane is possible; see [3] for a full proof and a testing procedure by MILP. In what follows we assume that this is not the case.

3. Necessary conditions for optimality

An instance x from a bag of G_{+1} (G_{-1}) is an *active point* for the solution (x_0, r_{+1}, r_{-1}) if $\|x - x_0\| = r_{+1}$ (r_{-1}). Given a solution (x_0, r_{+1}, r_{-1}), A_{+1} (A_{-1}) denotes the set of active points of G_{+1} (G_{-1}).

Theorem 1. *For any optimal solution (x_0, r_{+1}, r_{-1}) of Problem (3) some instances $a \in A_{+1}$ and $b \in A_{-1}$ exist.*

We then have $r_{+1} = \|a - x_0\| = \max_{j\in G_{+1}} \min_{x\in X_j} \|x - x_0\|$ and $r_{-1} = \|b - x_0\| = \min_{k\in G_{-1}} \min_{x\in X_k} \|x - x_0\|$.

Proof. Suppose that A_{+1} is empty. By feasibility of (x_0, r_{+1}, r_{-1}) some x_j from each X_j of G_{+1} satisfies $\|x_j - x_0\|^2 < r_{+1}^2$. Define then $r'_{+1} = \max_{x\in(\cup_{j\in G_{+1}} X_j)\cap B(x_0, r_{+1})} \|x - x_0\| < r_{+1}$, yielding the better feasible solution (x_0, r'_{+1}, r_{-1}).

In case $A_{-1} = \varnothing$, $\|x - x_0\|^2 > r_{-1}^2$ holds for any x belonging to any bag of G_{-1}. Therefore $r'_{-1} = \min_{x\in\cup_{i\in G_{-1}} X_i} \|x - x_0\| > r_{-1}$, and (x_0, r_{+1}, r_{-1}') improves the objective.

The final equalities follow directly from the feasibility of (x_0, r_{+1}, r_{-1}) and the definition of active points. □

Theorem 2. *For any optimal solution (x_0, r_{+1}, r_{-1}) we have*

1. *If $r_{+1} < r_{-1}$ there are at least two active points in G_{-1}.*
2. *If $r_{+1} > r_{-1}$ there are at least two active points in G_{+1}.*
3. *If $r_{+1} = r_{-1}$ and no instance is common to some positive and some negative bag, then there are at least two active points in G_{+1} and two in G_{-1}.*

Proof. We work again by contradiction by improving on (x_0, r_{+1}, r_{-1}) when the conditions are not satisfied.

1. When $r_{+1} < r_{-1}$, suppose that there is only one active point a in the set A_{-1} (see Theorem 1). Then the objective increases in direction $p = x_0 - a$. Indeed, if we move x_0 an amount $\epsilon > 0$, small enough (for not touching new active points) in the direction $u = p/\|p\|$, one has $x'_0 = x_0 + \epsilon u$. We may take as feasible solution (x'_0, r'_{+1}, r'_{-1}), where $r'_{-1} = r_{-1} + \epsilon$, while r'_{+1} may be taken as the maximum distance from x'_0 to the points of A_{+1}.
   For any $b \in A_{+1}$ we have by triangle inequality on b, x_0 and x_0' that $r_{+1}' \le r_{+1} + \epsilon$ and it follows that $r'^2_{-1} - r'^2_{+1} \ge (r_{-1} + \epsilon)^2 - (r_{+1} + \epsilon)^2 > r_{-1}^2 - r_{+1}^2$.
2. When $r_{+1} > r_{-1}$, suppose now there is only one active point b in the set A_{+1} and consider the direction $q = b - x_0$. If we move x_0 an amount $\epsilon > 0$, small enough, in direction $v = q/\|q\|$, one has that $r'_{+1} = r_{+1} - \epsilon$, while r'_{-1} may be taken as the distance of $x'_0 = x_0 + \epsilon v$ to closest point $a \in A_{-1}$. In a similar way as above we obtain that (x'_0, r'_{+1}, r'_{-1}) improves the objective.
3. In the case $r_{+1} = r_{-1}$ both arguments above apply with a small adaptation. First we observe that by the additional assumption the active points a and b cannot coincide. Then in case x_0, a and b would be colinear, the equality $r_{+1} = r_{-1}$ indicates that x_0 would be the midpoint between a and b, so the indicated move certainly improves. Finally when x_0, a and b are not colinear the needed triangle inequality is strict and the result still follows. □

Remark 1. From here on we assume that the instances of Ω are in general position: any set of n instances of Ω $(1 \le n \le d+1)$ is affinely independent.

Note that Theorem 2 then fully applies.

Theorem 3. *Any optimal solution (x_0, r_{+1}, r_{-1}) has at least $d+2$ active points.*

Proof. Let $A_{+1} = \{a = x_{j_1}, \ldots, x_{j_s}\}$ and $A_{-1} = \{b = x_{k_1}, \ldots, x_{k_t}\}$. By Theorem 1 $s, t \ge 1$. Consider the mediatrices of these two sets of active points

$$med(A_{+1}) = \{x \in \mathbb{R}^d : \|x - a\|^2 = \|x - x_{j_2}\|^2, \ldots, \|x - a\|^2 = \|x - x_{j_s}\|^2\}$$
$$med(A_{-1}) = \{x \in \mathbb{R}^d : \|x - b\|^2 = \|x - x_{k_2}\|^2, \ldots, \|x - b\|^2 = \|x - x_{k_t}\|^2\}$$

Then x_0 is a point of their intersection, and each point of this intersection within a small enough neighborhood of x_0 will have the same active points.

The intersection $med(A_{+1}) \cap med(A_{-1})$ is the solution set of the system of equations

$$\|x - a\|^2 = \|x - x_{j_i}\|^2 \quad (i = 2, \ldots, s)$$
$$\|x - b\|^2 = \|x - x_{k_h}\|^2 \quad (h = 2, \ldots, t)$$

which is equivalent to the linear system

$$2(x_{j_i} - x_a)^\top x = \|x_{j_i}\|^2 - \|x_a\|^2 \quad (i = 2, \ldots, s)$$
$$2(x_{k_h} - x_b)^\top x = \|x_{k_h}\|^2 - \|x_b\|^2 \quad (h = 2, \ldots, t)$$

which contains at most $s - 1 + t - 1$ linearly independent equations, so $dim(med(A_{+1}) \cap med(A_{-1})) \ge d - (s - 1 + t - 1) = d + 2 - (s + t)$.

In case there would be less than $d+2$ active points we would have $s + t \le d + 1$ so this intersection would have dimension ≥ 1 and thus contain some $y \ne x_0$ apart from x_0, which evidently lies

within it, and then also contains the whole line $L=\{x_0+\lambda(y-x_0)|\lambda\in\mathbb{R}\}$.

For the orthogonal projections a_0 and b_0 of the points a and b respectively on L we have

$$\begin{aligned} r_{-1}^2-r_{+1}^2 &= \|x_0-a\|^2-\|x_0-b\|^2 \\ &= \|x_0-a_0\|^2+\|a_0-a\|^2-\|x_0-b_0\|^2-\|b_0-b\|^2 \\ &= 2(b_0-a_0)^\top x_0 + C, \end{aligned} \tag{4}$$

where $C=\|a_0\|^2-\|b_0\|^2+\|a_0-a\|^2-\|b_0-b\|^2$ is a constant depending only on a, b, a_0 and b_0.

In case $a_0\neq b_0$ we can move x_0 to $x'_0=x_0+\epsilon(b_0-a_0)$ with $\epsilon>0$ small enough to keep the same active points and obtain as new value of the objective function

$$\begin{aligned} r'^2_{-1}-r'^2_{+1} &= 2(b_0-a_0)^\top(x_0+\epsilon(b_0-a_0))+C = r_{-1}^2-r_{+1}^2+2\epsilon\|b_0-a_0\|^2 \\ &> r_{-1}^2-r_{+1}^2. \end{aligned}$$

so would yield strictly better objective value.

If by accident $a_0=b_0$, we may always make another choice of $a'\in A_{-1}$ and $b'\in A_{+1}$ with different orthogonal projections. Indeed, if this was not possible the $d+1$ active points would lie on the same hyperplane normal to L, contrary to the general position assumption. □

Remark 2. Without the general position assumption, one still obtains existence of an optimal solution with at least $d+2$ active points (although the uniqueness is not guaranteed, since other solutions with the same value of the objective function and only $d+1$ active points can be found).

Proof. Indeed, if $dim(med(A_{+1})\cap med(A_{-1}))=1$ and the $d+1$ active points are cohyperplanar, their orthogonal projections on r will coincide. In that case, since $a_0=b_0$, expression (4) remains as follows:

$$r_{-1}^2-r_{+1}^2=\|a_0-a\|^2-\|b_0-b\|^2$$

which does not depend on x_0, therefore, we can move x_0 along the straight line until a new point becomes active and the value of the objective function remains constant. Then, a solution (x'_0, r'_{+1}, r'_{-1}) with $d+2$ active points can be reached, although the solution is not unique, because any solution with x_0^* belonging to the interval $[x_0, x'_0]$ would have the same value of the objective function (and only $d+1$ active points). □

Definition 1. A point x_0 is said to be generated by the sets $A^+\subset\cup_{j\in G_{+1}}X_j$ and $A^-\subset\cup_{i\in G_{-1}}X_i$ when $card(A^+\cup A^-)=d+2$, $med(A^+)\cap med(A^-)=\{x_0\}$ and all points of A^+ and A^- are active at x_0.

Theorem 3 asserts that any optimal solution is generated by its active points.

Theorem 4. *For any optimal solution (x_0, r_{+1}, r_{-1}) generated by A^+ and A^-, all points of A^+ come from different bags.*

Proof. By Theorem 2 we have $d+2$ active points for solution (x_0, r_{+1}, r_{-1}). In case A^+ contains two active points of the same bag we may drop one of these instances from this bag and obtain a new problem with less possible choices for the representatives of this bag. Therefore it should have an optimal value not larger than the original one.

However, for this new problem the current solution has the same objective value but only $d+1$ active points, and by Theorem 2 this cannot be optimal, hence a larger optimal value should exist, which is a contradiction. □

It has also been shown in [3] that the two groups of active points at any optimal solution have intersecting convex hulls. But this property cannot easily be used in an algorithm, so we do not develop it here.

4. A polynomial algorithm in fixed dimension

The results obtained in the previous section point toward the following full enumeration algorithm that is polynomial in fixed dimension d.

Algorithm 1. For all possible choices for the $d+2$ active points partitioned into sets A_{+1} and A_{-1} and satisfying the conditions obtained in Theorems 1–4, do

1. Compute the associated center and two radii.
2. Check the feasibility of the solution.
3. When feasible evaluate the solution and update the incumbent if necessary.

The center x_0 is built as the intersection of the mediatrices of the two sets of active points, A_{+1} and A_{-1}. Similarly as in the proof of Theorem 3 this may be done by fixing some $a_0\in A_{+1}$ and $b_0\in A_{-1}$ and solving the system of d linear equations

$$2(a-a_0)^\top x=\|a\|^2-\|a_0\|^2 \quad (a\in A_{+1}\setminus\{a_0\}) \tag{5}$$

$$2(b-b_0)^\top x=\|b\|^2-\|b_0\|^2 \quad (b\in A_{-1}\setminus\{b_0\}) \tag{6}$$

and should have a unique solution according to (Theorem 3). By Gauss elimination this has complexity $\mathcal{O}(d^3)$.

The radii are then computed in constant time by $r_{+1}=\|x_0-a_0\|$ and $r_{-1}=\|x_0-b_0\|$, which also yield the objective value $r_{-1}^2-r_{+1}^2$ when feasible.

Checking feasibility of this solution means checking $\exists x\in X_j: \|x-x_0\|^2\le r_{+1}^2$ for all bags of G_{+1} and $\|x-x_0\|^2\ge r_{-1}^2$ for all $x\in\cup_{k\in G_{-1}}X_k$, so takes $\mathcal{O}(n)$ where n is the total number of instances in the problem.

These steps must be repeated for the $\mathcal{O}(n^{d+2})$ possible choices of the active sets, leading to an overall complexity of $\mathcal{O}(\max(d^3,n)n^{d+2})$.

5. A VNS strategy

In high dimensions the complete enumeration algorithm is out of question due to its complexity. But in fact we are not really interested in obtaining an exact optimal solution of Problem (3) but rather in obtaining competitive results for the classification problem on the test data, and this may also be obtained with close to optimal solutions.

Therefore we propose here a Variable Neighborhood Search method (VNS), see e.g. [11] for a description. As shown there this metaheuristic has been successfully applied to many combinatorial and mixed-integer nonlinear optimization problems in many fields, among which also has classification (e.g. [4,22]).

In order to adapt VNS to our problem we now describe the search space, an initial solution, the neighborhood structure and the local search used for performing the algorithm. The algorithm is usually repeated many times yielding new opportunities for finding better solutions even with a fixed initial solution because of built-in choice randomization.

5.1. Search space

The finite search space consists of all pairs of sets A^+ and A^- satisfying

- Theorems 1 and 3: both nonempty and having together $d+2$ points.
- Theorem 4: A^+ selects at most one point from each bag of G_{+1}.
- A^- consists of instances of bags in G_{-1}.

5.2. Initial solution

In some experiments we used a random initial solution. However, a more sophisticated procedure to construct an initial solution is the following.

- Compute the centroid C_i of each bag i in G_{+1}, and then the centroid C of all C_i.
- Choose for each bag in $j \in G_{+1}$ the instance x_j closest to C.
- Select $d+2$ active points as follows:
  - for $s = \min(card(G_{+1}), d)$ choose the s points x_j $(j \in G_{+1})$ farthest from C,
  - choose the $d-s+2$ instances from $\cup_{k \in G_{-1}} X_k$ closest to C.

5.3. Neighborhood structure

The kth neighborhood $\mathcal{N}_k(A^+, A^-)$ consists of all possible pairs obtained by modifying k elements of the configuration (A^+, A^-) and belonging to the search space.

5.4. Local search

Given (A^+, A^-) the following local search steps are used to improve upon it.

1. The center x_0 is computed as the solution of the linear systems (5) and (6).
   In case the solution of the system is not unique the current solution is not a valid one, since according to the proof of Theorem 3 there must then exist more active points. We therefore (possibly repeatedly) add a new (random) active point (and consequently, a new equation is added to the linear system) until a unique solution is obtained.
2. The corresponding radii are then calculated according to Theorem 1 by

$$r_{+1} = \max_{j \in G_{+1}} \min_{x \in X_j} \|x - x_0\|$$
$$r_{-1} = \min_{k \in G_{-1}} \min_{x \in X_k} \|x - x_0\|$$

   and the corresponding (new) sets of active points

$$A^{+1} = \left\{ x \in \bigcup_{j \in G_{+1}} X_j \,\middle|\, \|x - x_0\| = r_{+1} \right\}$$
$$A^{-1} = \left\{ x \in \bigcup_{k \in G_{-1}} X_k \,\middle|\, \|x - x_0\| = r_{-1} \right\}$$

   are obtained.
   This construction guarantees at least one active point in each group.
3. If (A^{+1}, A^{-1}) belongs to the search space, i.e., satisfies all conditions of Section 5.1, the local search stops.
4. Otherwise we select $d+2$ active points as in the construction heuristic in Section 5.2, but based on x_0 instead of C:
   - For each bag in G_{+1}, take as its representative the instance of the bag that is closest to x_0 .
   - Select as A^{+1} the $s = \min(card(G_{+1}), d)$ representatives of bags in G_{+1} that are farthest from x_0,
   - Select as A^{-1} the $d-s+2$ instances in $\cup_{k \in G_{-1}} X_k$ closest to x_0.

   and restart from step 1.

Remark 3. Observe that during the process the value of r_{+1} cannot increase and r_{-1} cannot decrease, while when both remain equal the process stops. Therefore the objective value will always strictly increase, meaning that no cycling may occur. By finiteness of the search space this guarantees convergence. But the process might be quite long, so in practice a maximum number of iterations is fixed.

5.5. Main step of the algorithm

Given (A^+, A^-) in the search space with corresponding solution x_0, we choose at random a feasible pair $(A^{+'}, A^{-'}) \in \mathcal{N}_k(A^+, A^-)$, for $k=1$, on which we apply the local search procedure to end up with the new pair $(A^{+''}, A^{-''})$ and solution x''_0.

Now, we evaluate the objective function at x''_0. If it is better than x_0, we set $(A^+, A^-) = (A^{+''}, A^{-''})$ and restart the whole process. Otherwise we choose another random $(A^{+'}, A^{-'})$ from the same neighborhood and we repeat the process until having selected a maximum number of h solutions in each neighborhood.

After h iterations in a neighborhood, if the solution has not improved, we set $k := k+1$ and we continue the search in $\mathcal{N}_k(A^+, A^-)$, until $k = k_{max}$, where k_{max} is fixed to $d+2$ in our problem, at which point the neighborhood equals the whole search space. This whole process is repeated a fixed number of times.

Remark 4. The stopping rule may also be chosen by a maximum on the total number of iterations that depends on the problem size, instead of the dimension.

6. VNS algorithm for extended problems

In this section we discuss two extensions of the classification method and problem. First we consider the use of several separating balls instead of just a single one. Second we consider how to tackle problems of classifying bags into more than two classes.

6.1. The p-balls VNS algorithm

In most real databases, the value of the objective function for the solution of Problem (3) is negative, because one cannot construct the two separating concentric balls satisfying the constraints in Problem (3) and satisfying simultaneously that $r_{+1} \le r_{-1}$. In that case, one obtains a high misclassification rate for the training sample, when trying to separate the two groups, and may consequently expect bad results for classification in the test sample.

By allowing more than one separating ball one may try to improve these results using the following adaptation of the initial classification rule (1) to p balls, each ball $l = 1, \ldots, p$ with center $x_{0,l} \in \mathbb{R}^d$ and radius $r_l \in \mathbb{R}_+$. The MI assumption then becomes

Given a bag $X \subset \mathbb{R}^d$

$$\begin{aligned} &\bullet \text{ classify in } G_{+1}, && \text{if } \exists x \in X,\ \exists l \in \{1, \ldots, p\} \text{ such that } \|x - x_{0,l}\|^2 < r_l^2 \\ &\bullet \text{ classify in } G_{-1}, && \text{otherwise, i.e., if } \forall x \in X,\ \forall l \in \{1, \ldots, p\},\ \|x - x_{0,l}\|^2 \ge r_l^2. \end{aligned} \tag{7}$$

The heuristic is based on the following k-means clustering algorithm (see e.g. [12]), with $k = p$, to build clusters with the bags in G_{+1}:

1. Compute the centroid C_i of each bag i in G_{+1}.
2. Initial assignment: the set of bags is partitioned at random in p clusters (of roughly the same size).
3. Compute the centroid $\tilde{C}_l$ of each cluster l, $l = 1, \ldots, p$, as the mean of the centroids C_i of the bags assigned to that cluster.

4. Compute the distances between the centroid C_i of the ith bag, $i \in G_{+1}$, and the centroid $\tilde{C}_l$ of the lth cluster, $l=1,\ldots,p$.
5. Assign each bag i to the cluster l whose centroid $\tilde{C}_l$ is closest to C_i.
6. Repeat steps 3–5 while there are some changes in the assignment or while a fixed number of iterations is not reached.

Once the clusters have been constructed, we apply the VNS algorithm described in Section 5 within each cluster l, $l=1,\ldots,p$, i.e., using the set $G_{+1,l}$ of bags of the cluster l and $G_{-1,l}=G_{-1}$, to compute each of the p balls.

6.2. Multi-class case

So far, we have only dealt with the classification problem for two classes. However, in many real situations, more than two classes appear in a classification problem. Different strategies can be found in the literature, most of them proposing to transform the multi-class problem in a series of two-class problems (see e.g., [13,15,23]), as we also do here.

We will use the *one-versus-one* algorithm (1-v-1) for our experiments. In this algorithm, one has to construct a classifier for every possible pair of groups i and j. Since our classification rule is not symmetric, we will need to build for every pair (i,j) of classes the ball $B(x_0^{i,j}, r^{i,j})$ separating class i as G_{+1} from class j as G_{-1}, as well as the ball $B(x_0^{j,i}, r^{j,i})$ the other way round. In total we thus need to construct $N(N-1)$ two-class classifiers.

The classification rule for a bag X is then obtained by the following voting process:

1. For every pair of classes (i,j), we compute

$$intensity(i,j) = \frac{\min_{x \in X} d(x, x_0^{i,j})^2}{(r^{i,j})^2}, \tag{8}$$

2. Class i receives a vote for each j for which $intensity(i,j) < intensity(j,i)$.
3. Max Wins rule: In case the maximum votes are reached for two groups i and j, we assign the bag X to group i if $intensity(i,j) < intensity(j,i)$ (else, to group j).

7. Computational experiments

The classification problem and the VNS algorithm proposed for building the classifier have been implemented in Matlab 6.5 on a computer with Pentium IV CPU 3.06 GHz. The two-class method was tested on several artificial sets of data while the multi-class method was tested on a real database, as described below.

We always evaluated the classification method through 10-fold cross validation, that is, the bags of the database are partitioned in 10 sets and, in turn, 9 of these sets are used for training the model and the last one is used to test the problem. Thus, the process is repeated ten times (see [14,16] for a description of the method).

With the training sample we applied the VNS algorithm to build the center x_0 and the radius r which define the classifier, and we have measured the classification accuracy, i.e., the percentage of well-classified bags, first for the training sample itself and later for the test sample. The parameter k_{max} in the VNS algorithm has been fixed to $d+2$, while the parameter the number h of different solutions taken in each neighborhood and the maximum number of iterations in the local search step (see Section 5.4) have both been set to 5.

7.1. Full enumeration vs VNS algorithm

In this first experiment, we compare the results obtained with the VNS algorithm with those obtained with the full enumeration Algorithm 1 in the optimization problem (3).

Since full enumeration is not an efficient way of obtaining solutions in high dimension, two sets G_{+1} and G_{-1} have been built in dimension $d=2$ with 50 instances in each. Polar coordinates have been used for generating the instances. Thus, for an instance $(\rho\cos\theta, \rho\sin\theta)$ of G_{-1}, θ comes from a uniform distribution $U(-\pi,\pi)$, and ρ is chosen from a uniform distribution $U(r_1,r_2)$, where $0<r_1<r_2$ (r_1 was fixed to 1 and r_2 to 2 for the experiment). The instances in G_{+1} were generated in the same way, except for one instance in each bag that was included in $B(0,r_1)$ (ρ was uniformly distributed $U(0,r_1)$). In this way the spherical separability of the bags and a positive optimal value is guaranteed.

The experiment was repeated with 50 instances in each group and variable number of bags (1, 2, 5, 10 and 50) in the group G_{+1}, all bags with the same number of instances. Observe that the problem remains the same by changing the number of bags in G_{-1}.

In Table 1, we show the values of the objective function obtained for Problem (3) by using a complete enumeration and the VNS algorithm, for 1000 iterations, with the two possible initial solutions described in Section 5.2 (random and heuristic centroid-based initial solutions).

One can observe that the VNS algorithm obtains quite similar results to those obtained via a full enumeration, excepting the case of only one and two bags in G_{+1} where it even obtained a better value! However, in these cases it turned out that the problem was MI-separable by hyperplane, so the actual optimal solutions were unbounded, which is not detected by the complete enumeration that checks only candidates for spherical separation.

7.2. Spherically separable sets of instances

In this experiment we built another artificial database where the instances of G_{+1} and G_{-1} are spherically separable. For each group, 2000 instances have been generated: the instances of G_{+1} from a uniform distribution $U(-10,10)$ and the instances of G_{-1} from a uniform distribution $U(-20,20)$ but taking into account that at least one coordinate does not belong to the interval $[-10,10]$. The instances of each class were then grouped into 100 bags, 20 instances per bag.

This was repeated for dimensions $d=2,3,10,20,30,50,100$.

In all the cases the VNS method obtained an accuracy of 100% in each group for the training and the test sample in every dimension.

7.3. Spherically separable sets of bags

Another separable artificial database for the classification problem was constructed as follows. For each class (G_{+1} and G_{-1}), we generated a total of 2000 instances in 100 bags of 20 instances each. The instances of G_{-1} were generated as before with distribution $U(-20,20)$, with at least one coordinate not belonging to the interval $[-10,10]$. But now one instance of each bag of G_{+1} was generated uniformly $U(-10,10)$, while all remaining instances came from $U(-20,20)$.

The average accuracy for the 10 runs for different dimensions and for the test and training samples are given in Table 2. The number of iterations for the VNS algorithm, here without local search, was fixed to 2000.

One may observe that the accuracy remains quite satisfactory. Moreover, for the highest dimension ($d=100$), the accuracy is 100% in both cases (test and training sample). This is probably due to the fact that the higher the dimension, the more solutions are

Table 1
Value of the objective function for complete enumeration and for VNS.

Number of bags	1	2	5	10	50
Complete enumeration	1.9315	3.8101	0.3592	0.3036	0.0464
VNS (random initial solution)	1053.4	1664.6	0.3592	0.3036	0.0464
VNS (heuristic initial solution)	1053.4	1664.6	0.3592	0.3036	0.0464

Table 2
Accuracy (in %) for uniform artificial database.

Sample	Dim	$d=2$	$d=3$	$d=10$	$d=20$	$d=30$	$d=50$	$d=100$
Test	G_{+1}	100	92	93	97	97	97	100
	G_{-1}	100	90	87.69	93	97	99	100
	Total	100	91	90.35	95	97	98	100
Train	G_{+1}	100	92.22	96.44	99.78	99.89	100	100
	G_{-1}	100	95.57	78.86	98.56	100	100	100
	Total	100	93.9	87.65	99.17	99.95	100	100

Table 3
Accuracy (in %) for Gaussian artificial database.

Sample	Dim	$d=2$	$d=3$	$d=5$	$d=10$	$d=20$	$d=30$	$d=50$	$d=100$
Test	G_{+1}	96	88	97	100	100	100	100	100
	G_{-1}	87	89	96	100	99	100	100	100
	Total	91.5	88.5	96.5	100	99.5	100	100	100
Train	G_{+1}	100	99.89	99.67	100	100	100	100	100
	G_{-1}	87.56	89.33	97.45	100	100	100	100	100
	Total	93.78	94.61	98.56	100	100	100	100	100

considered in the VNS algorithm, since the k_{max}, that is, the maximum neighborhood radius taken into account, is fixed in our implementation to $d+2$. Hence, the number of solutions studied depends on the dimension of the problem.

7.4. Dataset based on a Gaussian distribution

For this database, 100 bags with 20 instances each, have been generated for each class G_{+1} and G_{-1}, by using a Gaussian distribution.

Each coordinate of the mean vector of each bag in G_{+1} comes from a uniform distribution $U(-1,1)$, while the coordinates of the mean vector of the bags in G_{-1} come from $U(-5,5)$. The instances of each bag are generated from a multivariate normal distribution with the corresponding mean vector and the identity as the covariance matrix.

Table 3 shows the average accuracy for the 10 runs of the cross-validation process in the test and training samples, for different dimensions (with 5000 iterations in the VNS algorithm which builds the classifier and without local search).

One can observe that the accuracy for the highest dimensions is better, in both samples (training and test), because for a high dimension, the databases built herein become more easily separable.

7.5. Real database for image categorization

Finally, we have applied our algorithm to a real database for image categorization. Image categorization consists in labeling images into a set of predefined categories.

The image database is a set of 2000 images in JPEG format taken from 20 CD-ROMs published by the COREL Corporation, each CD-ROM containing 100 images representing a similar concept. This dataset was previously used for Multi-instance Learning in [6,7], and it is available at the webpage http://www.cs.olemiss.edu/~ychen/ddsvm.html.

A segmentation process was applied to these images to extract some properties about luminance, color and texture of the pictures and they were encoded into feature vectors. These feature vectors were grouped into clusters, representing the regions of the segmented image. In this way each image has several regions, where each region is characterized by a feature vector in dimension $d=9$, representing the color, texture and shape properties of that region (see [6,7] for a more detailed description).

From the Multiple Instance Learning framework, the different concepts (CD-ROMs) are the classes to which the images are assigned, the images are the bags of the database, and the regions are the instances of each bag. In this dataset, there are 20 classes, 100 bags in each class and the average number of instances per bag for the different classes is displayed in Table 4 (along with the name of the classes). The dimension of the problem is $d=9$. We have performed several experiments with only the first 10 groups (1000-Image database) and with the complete database (2000-Image database).

Since the database has more than two classes, the 1-v-1 algorithm, explained in Section 6.2, is the selected tool to solve the multi-class problem, and for every pair of classes (i,j) the p-balls VNS algorithm, described in Section 6.1, is used to build the classifier.

First, we have considered the problem with only the first ten classes. For selecting the training and test samples, we have used 5-fold cross validation on the database. Different values for p (in the optimization algorithm to construct the separating balls) have been considered (from $p=1$ to $p=20$), although we only show the best results, which were obtained for $p=15$. The number of iterations to obtain each ball is set equal to 50, and the solution

based on the centroid (see Section 5.2) was taken as the initial solution.

Table 5 displays the confusion matrix for the test samples in the database with the first 10 classes. Each element (i,j) of this matrix represents the percentage of elements of the class i which has been assigned to the class j. The elements of the diagonal (in bold) represent the percentage of elements correctly labeled for each class. One may observe that the class 4 (dinosaurs) is easily separable from the other classes, since all its elements have been correctly classified. However, it turns out to be much harder to distinguish between classes 0 and 5 (African people or villages and elephants), or more particularly between the two kinds of landscapes: images of beaches and images of mountains or glaciers (classes 1 and 8), where we obtain 17% of beaches misclassified as mountains or glaciers, and 18% in the other direction. Likewise, 14% of horses (class 7) are labeled as elephants (class 5).

Table 4
Description of the image database.

Class	Class name	Instances per bag (average)
0	African people and villages	4.84
1	Beach	3.54
2	Historical building	3.1
3	Buses	7.59
4	Dinosaurs	2.00
5	Elephants	3.02
6	Flowers	4.46
7	Horses	3.89
8	Mountain and glaciers	3.38
9	Food	7.24
10	Dogs	3.80
11	Lizards	2.80
12	Fashion models	5.19
13	Sunset scenes	3.52
14	Cars	4.93
15	Waterfalls	2.56
16	Antique furniture	2.30
17	Battle ships	4.32
18	Skiing	3.34
19	Desserts	3.65

Table 8
Accuracy for the two databases.

Sample	1000-Image database	2000-Image database
Test	75.3	62.6
Train	90.45	89.24

Table 9
Accuracy for different algorithms for the image database.

Algorithms	1000-Image database	2000-Image database
MILES	82.6	68.7
DD-SVM	81.5	67.5
MI-SVM	74.7	54.6
k-means-SVM	69.8	52.3
***p*-balls VNS algorithm**	75.3	62.6

Table 5
Confusion matrix for the 1000-Image database.

	Assigned class									
Real class	0	1	2	3	4	5	6	7	8	9
0	**67**	3	4	0	2	11	1	3	3	6
1	2	**58**	7	3	1	8	1	0	17	3
2	4	3	**75**	2	1	7	1	0	5	2
3	1	3	10	**67**	9	2	0	1	1	6
4	0	0	0	0	**100**	0	0	0	0	0
5	10	4	2	0	0	**71**	0	5	8	0
6	2	0	0	0	0	1	**94**	2	0	1
7	3	1	0	0	0	14	0	**81**	1	0
8	0	18	3	0	0	10	0	0	**69**	0
9	5	4	1	2	5	4	0	3	5	**71**

Table 6
Accuracy for the 1000-Image database.

Class	0	1	2	3	4	5	6	7	8	9
Test	67	58	75	67	100	71	94	81	69	71
Train	83.75	78.5	90.5	96.5	100	86.25	99.5	96.25	75.5	97.75

Table 7
Accuracy for the 2000-Image database.

Class	0	1	2	3	4	5	6	7	8	9
Test	52	58	69	67	97	55	88	78	42	67
Train	84.25	83.5	91	93.25	100	83.25	98.75	92	68.75	95.25
Class	10	11	12	13	14	15	16	17	18	19
Test	44	63	64	57	57	76	81	60	49	28
Train	86	82.5	96	92.25	92	92.5	99.25	95.5	84	74.75

Table 6 shows the accuracy for every class, that is, the percentage of elements of every class that have been correctly labeled into that class, both within the training and the test samples. One can observe that the performance of the algorithm is quite good in most of the classes in the training sample, and, in general, a class which is easily separable from the rest in the training sample, continues being easily discriminated in the test sample. This is the case of class 4, with 100% of accuracy in both the training and the test samples. However, we can also find some classes, like class 3 (buses) with a much better accuracy in the training (96.5%) than in the test sample (only 67%).

Table 7 presents the classification accuracy for every class in the complete dataset (2000-Image database). The performance of our algorithm is good in the training sample in most of the classes (except for class 8), showing the power of our methodology to separate the bags of the different concepts. In the test sample, the accuracy is lower than for the 1000-Image database, although good results are obtained for separating classes such as number 4 and 6 (dinosaurs and flowers).

In Table 8, the accuracy for the test and training samples in the two databases are shown. One can observe that we obtain very good results for the training sample (even with the complete dataset), around 90%, and the results for the test sample are quite competitive.

Finally, in table 9 we compare the results we have obtained with the results obtained via other methods that have been used on this database: MILES algorithm [6], DD-SVM [7], MI-SVM [1] and *k*-means-SVM [9]. These other algorithms have also been tested on five test sets extracted at random from the database, but the technique is not cross validation (see [6]). Although our algorithm was not able to improve the best results obtained so far, our results are overall comparable with the solutions obtained for this database, and in fact improve on the performance of other algorithms based on SVMs (like MI-SVM and *k*-means-SVM) in this multi-class problem in both datasets.

Observe, however, that the *p*-balls VNS algorithm as presented here may be considered as rather simplistic, since it is based on a-priori clustering and the VNS method is applied on the fixed groups only. It should be possible to incorporate a dynamically changing clustering in an all-over VNS strategy that promises to give even better results. Such a method and its testing remains to be done.

Acknowledgments

The work of both first authors has been partially supported by the grants MTM2009-14039-C06-06 of MEC, Spain, and FQM-329 of Junta de Andalucía, Spain. Their financial support is gratefully acknowledged.

References

[1] Andrews S, Tsochantaridis I, Hofmann T. Support vector machines for multiple-instance learning. In: Proceedings of the advances in neural information processing systems, vol. 15; 2002. p. 561–8.
[2] Astorino A, Fuduli A, Gaudioso M. DC models for spherical separation. Journal of Global Optimization 2010;48:657–69.
[3] Carrizosa E, Gordillo J, Plastria F. Building separating concentric balls to solve a multi-instance classification problem. Optimization online 2008, paper 1897.
[4] Carrizosa E, Martin-Barragan B, Plastria F, Romero-Morales D. On the selection of the globally optimal prototype subset for NN classification. INFORMS Journal on Computing 2007;19:470–9.
[5] Carrizosa E, Romero Morales D. Supervised classification and mathematical optimization. Computers & Operations Research 2013;40:150–65.
[6] Chen Y, Bi J, Wang JZ. MILES: multiple-instance learning via embedded instance selection. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(12):1931–47.
[7] Chen Y, Wang JZ. Image categorization by learning and reasoning with regions. Journal of Machine Learning Research 2004;5:913–39.
[8] Cristianini N, Shawe-Taylor J. An introduction to support vector machines and other kernel-based learning methods. Cambridge: Cambridge University Press; 2000.
[9] Csurka G, Bray C, Dance C, Fan L. Visual categorization with bags of keypoints. In: Proceedings of the ECCV'04 workshop statistical learning in computer vision; 2004 p. 59–74.
[10] Dietterich TG, Lathrop RH, Lozano-Pérez T. Solving the multiple instance problem with axis-parallel rectangles. Artificial Intelligence 1997;89:31–71.
[11] Hansen P, Mladenović N, Moreno Pérez JA. Variable neighbourhood search: methods and applications. Annals of Operations Research 2008;6:319–60.
[12] Hartigan JA. Clustering algorithms. New York: John Wiley and Sons; 1975.
[13] Hastie T, Tibshirani R. Classification by pairwise coupling. Annals of Statistics 1998;26(2):451–71.
[14] Hastie T, Tibshirani R, Friedman J. The elements of statistical learning: data mining, inference, and prediction. New York: Springer-Verlag; 2001.
[15] Hsu C-W, Lin C-J. A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 2002;13(2):415–25.
[16] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proc. of the 14th international joint conference on artificial intelligence; 1995. p. 1137–43.
[17] Gabriel Krummenacher, Cheng Soon Ong, Joachim M. Buhmann, Ellipsoidal multiple instance learning. In: Proc. of the 30th international conference on machine learning, vol. 28. Atlanta, Georgia, USA: JMLR: W&CP; 2013.
[18] Erhun Kundakcioglu O, Seref Onur, Pardalos Panos M. Multiple instance learning via margin maximization. Applied Numerical Mathematics 2010;60:358–69.
[19] Li Yan, Tax David MJ, Duin Robert PW, Loog Marco. Multiple-instance learning as a classifier combining problem. Pattern Recognition 2013;46:865–74.
[20] Mangasarian OL, Wild EW. Multiple instance classification via successive linear programming. Journal of Optimization Theory and Applications 2008;137:555–68.
[21] Pao HT, Chuang SC, Xu YY, Fu H-C. An EM based multiple instance learning method for image classification. Expert Systems with Applications 2008;35:1468–72.
[22] Plastria F, De Bruyne S, Carrizosa E. Alternating local search based VNS for linear classification. Annals of Operations Research 2010;174:121–34.
[23] Schölkopf B, Smola A. Learning with kernels: support vector machines, regularization, optimization and beyond. Cambridge: MIT Press; 2002.
[24] Qi Z, Tian Y, Shi Y. Multi-instance classification based on regularized multiple criteria linear programming. Neural Computing and Applications First online 2012: 1–7. http://dx.doi.org/10.1007/s00521-012-1008-0.
[25] Zhou Z-H, Zhang M-L. Solving multi-instance problems with classifier ensemble based on constructive clustering. Knowledge and Information Systems 2007;11:155–70.