HyDR-MI: A Hybrid Algorithm to Reduce Dimensionality in Multiple Instance Learning

Amelia Zafra[a], Mykola Pechenizkiy[b], Sebastián Ventura[*,a]

[a]*Department of Computer Science and Numerical Analysis. University of Cordova*
[b]*Department of Computer Science. Eindhoven University of Technology*

Abstract

Feature selection techniques have been successfully applied in many applications for making supervised learning more effective and efficient. These techniques have been widely used and studied in traditional supervised learning settings, where each instance is expected to have a label. In multiple instance learning (MIL) each example or bag consists of a variable set of instances, and the label is known for the bag as a whole, but not for the individual instances it consists of. Therefore utilizing these labels for feature selection in MIL becomes less straightforward. In this paper we study a new feature subset selection method for MIL called HyDR-MI (Hybrid Dimensionality Reduction method for Multiple Instance learning). The hybrid consists of the filter component based on an extension of the ReliefF algorithm developed for working with MIL and the wrapper component based on a genetic algorithm that optimizes the search for the best feature subset from a reduced set of features, output by the filter component. We conducted an extensive experimental evaluation of our method on five benchmark datasets and seventeen classification algorithms for MIL. The results of our study show the potential of the proposed hybrid with respect to the desirable effect it produces: a significant improvement of the predictive performance of many MIL classification techniques as compared to the effect of filter-based feature selection. This is achieved due to the possibility to decide how many of the top ranked features are useful for each particular algorithm and the possibility to discard redundant attributes.

Key words: Multiple Instance Learning, Feature Selection, ReliefF Algorithm, Genetic Algorithm

1. Introduction

The estimation of the quality of the attributes (or extracted features) is of extreme importance in machine learning. In many applications there are hundreds or thousands of attributed describing each input object in the population. While many of these features are potentially useful for learning, lots of other features can be irrelevant, redundant or noisy. In this scenario, when

[*]Department of Computer Science and Numerical Analysis. University of Cordova. Campus Universitario Rabanales, Edificio Einstein, Tercera Planta. 14071 Cordoba. SPAIN. Tel:+34957212031; Fax:+34957218360

Email addresses: `azafra@uco.es` (Amelia Zafra), `m.pechenizkiy@tue.nl` (Mykola Pechenizkiy), `sventura@uco.es` (Sebastián Ventura) data have hundreds or thousands of features, the majority of learning methods do not perform well showing the drastic increase in computational complexity and classication error. From a statistical point of view, examples with many irrelevant and noisy features provide very little information. Therefore applying a feature selection method would reduce the dimensionality by discarding irrelevant features and present the learning algorithm only with those features that are really relevant to describe the target concept is highly desirable. In this way, an adequate selection of features may improve the accuracy and efficiency of classification methods. Besides, the feature selection process may bring additional benefits including data comprehension, gaining knowledge about the process, reducing data, limiting storage requirements and helping to generate simpler models and higher speeds.

Feature selection methods have been traditionally divided into three broad categories with respect to how they combine feature selection search with the construction of the classification model: filter methods, wrapper methods, and embedded methods (19). Although wrapper and embedded techniques get competitive results because they incorporate variable selection as part of the training process, they normally require more computation time. Filter methods on the contrary are computationally simpler and faster because they act independently of the model induction process but tend to result in worse accuracy of learnt models as compared to wrapper and embedded feature selection. To overcome the deficiencies of these methods when working independently, they have been combined in some studies. These combinations have produced excellent results (35; 28; 64; 25). However, all the proposed techniques that we are aware of have been studied in the traditional supervised learning setting, where each instance is expected to have a label. Although there have been numerous theoretical studies on the relationship between supervised and multiple instance learning (MIL) in different scenarios, and on the importance of considering special adaptations to appropriately solve problems based on this learning (32; 59; 61), reduction dimensionality in MIL has barely been dealt with. Utilizing class labels for feature selection in MIL is not straightforward because in MIL each example or bag consists of a variable set of instances, and the label is known for the bag as a whole, but not for the individual instances it consists of. The number of positive instances in the bag is usually also not known. Therefore, designing a simple filter-based feature selection for classification in MIL settings is tricky. Recently in (49) we have introduced a filter based approach for feature selection that called ReliefF-MI. It adapts the ideas of the popular ReliefF (23) algorithm to MIL.

In this paper we design a hybrid method of feature selection called HyDR-MI (*Hy*brid *D*imensionality *R*eduction method for *M*ultiple *I*nstance learning) to deal with MIL problems. This method combines the advantages of filter and wrapper methods. To determine the important properties of the feature space, a filter method is used which assigns a score (numerical value) to each attribute according to its importance for classification. Thus the features having very low scores are considered to be irrelevant and therefore can be discarded. The new, reduced set of features is provided to a wrapper whose purpose is to select the best feature subset for a particular MIL algorithm. Our hybrid uses ReliefF-MI as the filter method and a genetic algorithm (GA) in the wrapper to optimize the search for the best feature subset from the already reduced set of features by the filter. ReliefF-MI inherits the advantages of ReliefF which include: a low bias, accounting for interactions among the features and capturing local dependencies (that other methods may miss). GA is chosen as the search mechanism in the wrapper because its potential has been proven in the scientific and engineering optimization and it is naturally applicable to feature selection since the problem has an exponential search space. This combination benefits both sides, because on one hand, the main restriction of the ReliefF-MI algorithm is the necessity of setting a threshold determining how many top scored features to select. Besides, it evaluates each feature individually and therefore cannot handle well the problem of feature redundancy from which a classification technique may suffer. GA helps to search for the best feature subset addressing these issues. We use performance of a classifier as the fitness function in GA and this optimizes the feature selection for finding a subset most suitable for that particular classifier. And on the other hand, GA being computationally expensive benefits from performing the search in a smaller space consisting of the set of features provided by ReliefF-MI.

The results reported in several studies on feature selection in MIL (54; 47; 33) are scattered and not directly comparable to each other due to different (or incomplete descriptions of) experimental settings. For this reason, we conduct an extensive experimental study aimed to evaluate the effect of feature selection on the performance of the most popular MIL algorithms (seventeen algorithms in total) on five MIL benchmark datasets. Our experimental results reveal that HyDR-MI is a promising hybrid that improves the accuracy of a considerable number of classifiers, and which performance is superior to ReliefF-MI filter.

The remainder of the paper is structured as follows. Section 2 briefly introduces the MIL framework and related work on feature selection. Section 3 gives a detailed specification of HyDR-MI, the studied hybrid method for dimensionality reduction in MIL. Section 4 presents the experimental evaluation of different MIL techniques with (and without) feature selection. Finally, Section 5 draws conclusions and raises several issues for future work.

2. Background and Related Work

In this section, a brief definition of multiple instance learning is given, along with a description of related work.

2.1. Multiple Instance Learning

MIL is a special learning framework which deals with the uncertainty of instance labels. Thus, instead of receiving a set of instances which are labeled as positive or negative, the learner receives a set of bags labeled as positive or negative. Each bag can contain a different number of instances whose labels remain unknown. The goal of this learning paradigm is to learn a concept that correctly classifies the training data and generalizes to unseen data. Although the actual learning process is quite similar to traditional supervised learning, the main difference between the two approaches is found in the class labels provided by the data. According to the specification given by Dietterich et al. (12), in a traditional setting of machine learning, an object m is represented by a feature vector $V(m)$ which is associated to a label $f(m)$. However, in the multiple instance setting, each object m may have i various instances denoted $m_1, m_2, \ldots, m_i$. Each of these variants will be represented by a (usually) distinct feature vector $V(m)$. A complete training example is therefore written as $(\{V(m_1), V(m_2), \ldots, V(m_i)\}, f(m))$. In this case, the label $f(m)$ represents the information in the examples, not about each individual instance. The goal of learning is to find a good approximation to function $f(m)$, $\hat{f}(m)$, analysing a set of training examples and labeled by $f(m)$. To obtain this function Dietterich defines a hypothesis that assumes that if the result observed is *positive*, then at least one of the variant instances must have produced that positive result. Furthermore, if the observed result is *negative*, then none of the variant instances could have produced a positive result. Dietterich et al. (12) model this problem by introducing a second function $g(V(m_{i,j}))$ that takes a single variant instance and produces a result. The externally observed result, $f(m)$, can then be defined as follows:

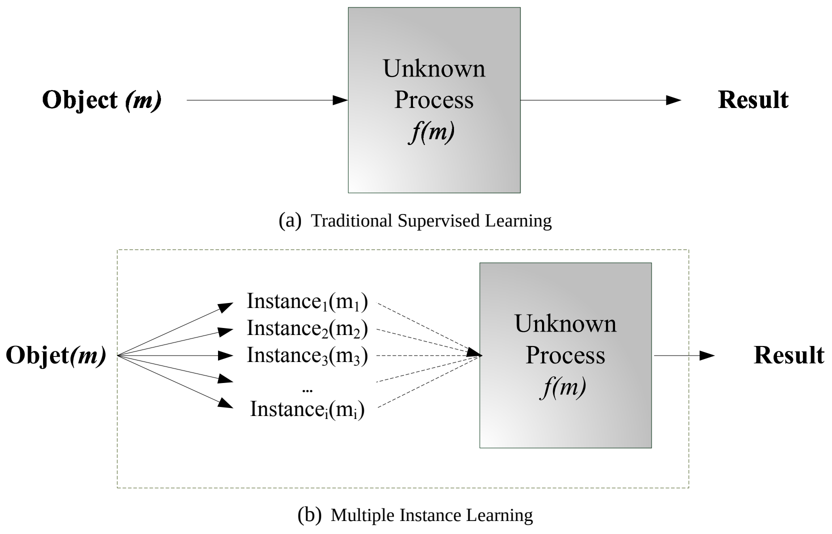


Figure 1: Differences between traditional supervised learning (with single instances) and multiple instance learning, (Dietterich et al. (12))

$$f(m_i) = \begin{cases} 1 & \text{if } \exists\, j \mid g(V(m_j)) = 1 \\ 0, & \text{otherwise} \end{cases}$$

From this initial definition, other hypotheses have been used in MIL (42; 37) that extend this assumption.

However, what makes this problem complex is the fact that there is no information about which of the instances are actually positive or their number, and this lack of information complicates the learning process. According to the notation specified, the difference between the two learning frameworks can be represented clearly in Figure 1 where the difference between the input objects is shown.

Learning with multi-instances has become widespread in the last few years due to the great number of applications that have found a more appropriate form of representation in MIL settings rather than in traditionally defined supervised learning. There are approaches for text categorization (2), content-based image retrieval (30; 20) and image annotation (45; 31), drug activity prediction (27; 63), web index page recommendation (60; 52), semantic video retrieval (6), video concept detection (18; 1), pedestrian detection (29), and the prediction of student performance (50). In all cases MIL provides a more natural form of representation, and is able to improve the accuracy results as compared to the performance of the traditional supervised learning.

In order to solve these problems, many new MIL methods have been designed. Some of these methods have been designed specifically for solving MIL problems. Popular approaches of this type include APR algorithms (12), Diverse Density (DD) (27), EM-DD (58) and, an algorithm by Pao et al. (30). Other methods are based on adapting popular machine learning paradigms to MIL settings. Algorithms belonging to this category include multi-instance lazy learning algorithms (41), multi-instance tree learners (36; 9; 4), multi-instance rule inducers (9), multi-instance logistic regression methods (44), multi-instance neural networks (54; 55; 56; 5), multi-instance kernel methods (7; 17; 8; 26), multi-instance ensembles (62; 63; 55) and evolutionary algorithms (52; 51; 48).

2.2. Feature Selection in Multiple Instance Learning

Although there is a lot of research in the area of traditional supervised learning that show the desirable properties of feature selection methods, it is very difficult to find specific applications in MIL. As previously mentioned, this learning paradigm is considered to be an extension of traditional supervised learning, which has generated an enormous amount of scientific activity involving the design of a considerable number of applications and algorithms. However, there are a very few studies on feature selection in MIL. The first study on this topic we are aware of was carried out by Zhang and Zhou (54) who improved one MIL method named BP-MIP by adopting two different feature selection techniques (feature scaling with Diverse Density and feature reduction with principal component analysis). In there work, feature vectors that are fed to a BP-MIP neural network have been previously scaled by the feature weights found by running Diverse Density on the training data, or have been projected by a linear transformation matrix formed by principal component analysis. Later, Yuan et al. (47) formulated region-based image retrieval as an MIL problem, and proposed an algorithm named MI-AdaBoost to resolve it. The algorithm firstly maps each bag into a new bag feature space using a certain set of instance prototypes, and then adopts AdaBoost to select the bag features and build classifiers simultaneously. Then, Raykar et al. (33) proposed a Bayesian MIL algorithm that uses feature selection. This algorithm automatically identifies the relevant feature subset, and uses inductive transfer when learning multiple (conceptually related) classifiers. Normally, the methods presented in the literature up til now include the process of feature selection within the algorithm that they propose and are introduced and compared to a few other approaches that do not use any feature selection methods.

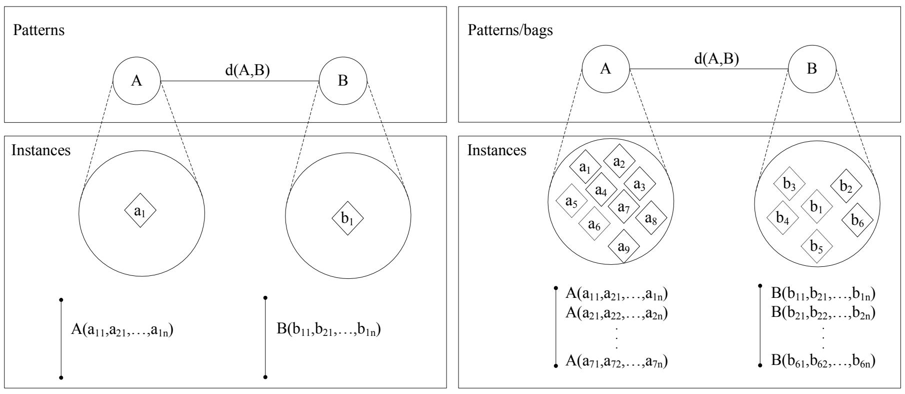
3. HyDR-MI: A Hybrid Dimensionality Reduction Method for Multiple Instance Learning

This section describes our hybrid approach for feature selection in MIL. First, we describe the filter component of the hybrid, which is based on an extension of the ReliefF algorithm for MIL, and then we describe how the filter component is integrated with the GA wrapper-based feature subset selection.

3.1. ReliefF-MI. Adaptation of ReliefF to Multiple Instance Learning

Relief algorithms are general and successful attribute estimators. They are commonly viewed as feature subset selection methods applied in a pre-processing step before the model is learned, and are among the most successful pre-processing algorithms.

Before specifying the algorithm proposed, its differences with respect to the ReliefF algorithm (23) are delimited in order to better understand how our method is developed. It is generally known that the original ReliefF algorithm estimates the quality of attributes by how well their values distinguish between patterns that are near to each other. In this context, a pattern consists of one instance. In MIL, the situation is different because the distance between two patterns has to be calculated by taking into account that each pattern can contain one or more instances. The gist of this type of learning is to make a decision based on similarity because each example is a set of instances, so the similarity function needs to be upgraded because the Manhattan distance measure schema is not applicable. The difference between the two cases



(a) Distance between patterns with single instances (b) Distance between patterns with multiple instances

Figure 2: Differences between ReliefF and ReliefF-MI

can be seen in Figure 2. This figure shows the calculation of the distance between two patterns or examples of data sets considering both traditional supervised learning and multiple instance learning. Figure 2(a) shows the calculation in a traditional supervised learning scenario. In this case, the correspondence between pattern and instance is one-to-one and a simple Manhattan distance could be carried out because it is only necessary to calculate the distance between two feature vectors. Figure 2(b) shows the case in MIL where the correspondence between pattern and instance is one-to-many, therefore the distance between the two patterns is different. In this case, the pattern, most commonly called a bag, has seven different instances in the case of A and six different instances for bag B (in this learning each bag may contain a different number of instances). Therefore, the Manhattan distance is not possible in this scenario and other suggestions have to be made.

The literature proposes different distance-based approaches to solve MI problems (9; 53; 57). The most extensively used metric is the Hausdorff distance (14), which measures the distance between two sets. To evaluate its performance, four adaptations of this measurement have been implemented: maximal Hausdorff, minimal Hausdorff, average Hausdorff (used previously in MIL), and a new metric proposed by the authors called adapted Hausdorff. Thus, four different versions of ReliefF-MI which use these metrics are implemented. In each alternative we follow the same general steps shown in Listing 1 and they are based on the ReliefF method (23; 21). The main difference between the different alternatives is the definition of the function *diff* which is responsible for calculating the difference between two bags for a given attribute. Moreover, it is also used to calculate the distance between bags to find the nearest neighbours.

In continuation, the metric and differential function in different versions is shown. In all cases R_i is considered as the bag selected in the current iteration which contains three instances (R_i^1, R_i^2, R_i^3), H_j is the j^{th} bag of the k nearest hit selected in the current iteration which contains four instances, $(H_j^1, H_j^2, H_j^3, H_j^4)$, and M_j is the j^{th} bag of the k nearest misses selected in the current iteration which contains six instances, $(M_j^1, M_j^2, M_j^3, M_j^4, M_j^5, M_j^6)$.

ReliefF-MI with maximal Hausdorff distance. This extension of ReliefF for MIL uses the *maximal Hausdorff distance* (14). This distance is the classical Hausdorff distance and can

Listing 1 – ReliefF-MI Method (23)

```
Set selectedSubset = $\emptyset$.
Set W = 0.
Set m = value specified by user, representing the number of times that the process is repeated.
for i=1 to numberExamples do
    Get one bag $R_i$ from the training data set D.
    Get $k$ nearest hit : $H_1, \ldots, H_k$ ($H_j$ bag in D where dist($R_i$,$H_j$) is $j^{th}$ closest & $R_i$.class=$H_j$.class)
    for each class C <> Class ($R_i$) do
        Get $k$ nearest miss $M_1, \ldots, M_k$ ($M_j$ bag in D where dist($R_i$,$M_j$) is $j^{th}$ closest & $R_i$.class<>$M_j$.class )
    end for
    for A=1 to numberFeatures do
        Set $W[A] = W[A] - \frac{\sum_{j=1}^{k} diff(A, R_i, H_j)}{m \cdot k} + \sum_{C \neq Class(R_i)} \left[ \frac{\frac{P(C)}{1-P(class(R_i))} \sum_{j=1}^{k} diff(A, R_i, M_j(C))}{m \cdot k} \right]$
    end for
end for
for i=1 to numberFeatures do
    if W[i] > threshold then
        Add feature $i$ to selectedSubset.
    end if
end for
```

be specified as:

$$H_{max}(R_i, H_j) = max\{h_{max}(R_i, H_j),\ h_{max}(H_j, R_i)\}$$
$$\text{where } h_{max}(R_i, H_j) = max_{r \in R_i} min_{h \in H_j} ||r - h||$$

To calculate the difference between the attributes of two patterns, the instance selected is the maximal distance of the minimal distance between the different instances of one bag and another: $diff_{bag-max}(A, R_i, H_j) = diff_{instance}(A,\ R_i^3,\ H_j^4)$ where R_i^3 and H_j^4 are instances which satisfy this condition. Similarly, $diff_{bag-max}(A, R_i, M_j)$ is computed but with the instances of R_i and M_j.

ReliefF-MI with minimal Hausdorff distance. This extension of ReliefF for MIL uses the *minimal Hausdorff distance (9)*. Instead of choosing the maximum for this distance, the distance is ranked first and the lowest distance value is selected. Formally, it modifies the definition of the Hausdorff distance as follows:

$$H_{min}(A, B) = min_{a \in A} min_{b \in B} ||a - b||$$

To calculate the difference between the attributes of two patterns, the instance of each bag selected to calculate the distances is the minimal distance between all instances of each bag: $diff_{bag-min}(A, R_i, H_j) = diff_{instance}(A, R_i^1, H_j^3)$, where R_i^1 and H_j^3 are instances that satisfy this condition. Similarly, $diff_{bag-min}\ (A, R_i, M_j)$ would be computed but with the instances of R_i and M_j.

ReliefF-MI with average Hausdorff distance. This extension of ReliefF for MIL uses the *average Hausdorff distance (57)* proposed by Zhang and Zhou to measure the distance between two bags. It is defined as follows:

$$H_{avg}(R_i, H_j) = (\sum_{r\in R_i} min_{h\in H_j}||r-h|| + \sum_{h\in H_j} min_{r\in R_i}||h-r||)/(|R_i| + |H_j|)$$

where |.| measures the cardinality of a set. $H_{avg}(A, B)$ averages the distances between each instance in one bag and its nearest instance in the other bag. Conceptually speaking, average Hausdorff distance takes more geometric relationships between two bags of instances into consideration than the maximal and minimal Hausdorff ones do.

To calculate the difference between the attributes of two patterns, there are several instances involved for updating the weights of the features. Supposing

- $d(R_i^1, H_j^2)$, $d(R_i^2, H_j^1)$ and $d(R_i^3, H_j^4)$ are the minimal distance between each instance $r \in R_i$ with respect to instances $h \in H_j$.
- $d(H_j^1, R_i^1)$, $d(H_j^2, R_i^1)$, $d(H_j^3, R_i^2)$ and $d(H_j^4, R_i^3)$, are the minimal distances between each instance $h \in H_j$ with respect to the instances $r \in R_i$.

The function *diff* would be specified as follows,

$$diff_{bag-avg}(A, R_i, H_j) = \frac{1}{r+h} * [diff_{instance}(A, R_i^1, H_j^2) + diff_{instance}(A, R_i^2, h_j^1)$$
$$+ diff_{instance}(A, R_i^3, h_j^4) + diff_i nstance(A, H_j^1, R_i^1) + diff_{instance}(A, H_j^2, R_i^1)$$
$$+ diff_{instance}(A, H_j^3, R_i^2) + \ diff_{instance}(A, H_j^4, R_i^3)]$$

The same process is used to calculate $diff_{bag-avg}(A, R_i, M_j)$, but considering the pattern M_j.

ReliefF-MI with adapted Hausdorff distance. This extension of ReliefF for MIL uses the *adapted Hausdorff distance*. Due to the particularities of this learning, this new distance is proposed combining the previous ones to measure the distance between two bags. This metric represents a different calculation depending on the class of the pattern because the information about instances in each pattern depends on the class that it belongs to. Thus, the metric is different if we evaluate the distance between two positive or negative patterns or that between one positive and one negative pattern.

- If both patterns are negative, we can be sure that there is no instance in the pattern that represents the concept we want to learn. Therefore, an average distance will be used to measure the distance between these bags because all instances are guaranteed to be negative: $H_{adapted}(R_i, H_j) = H_{avg}(R_i, H_j)$.
- If both patterns are positive, the correct information is that at least one instance in each of them represents the concept that we want to learn, but there is no information about which particular instance or set represents the concept. Therefore, the minimal distance is used to measure their distance because the positive instance has more probability of being near: $H_{adapted}(R_i, H_j) = H_{min}(R_i, H_j)$
- Finally, if we evaluate the distance between patterns where one of them is a positive bag and the other is a negative one, the measurement considered will be the maximal Hausdorff distance because the instances in the different classes are probably outliers between the two patterns: $H_{adapted}(R_i, M_j) = H_{max}(R_i, M_j)$

In this case, the calculation of the *diff* function also depends on the pattern class. Therefore, the pattern label will determine how the function *diff* will be evaluated.

- If R_i is positive and H_j is positive,

$$diff_{bag-adapted}(A, R_i, H_j) = diff_{bag-min}(A, R_i, H_j)$$

- If R_i is negative and H_j is negative,

$$diff_{bag-adapted}(A, R_i, H_j) = diff_{bag-avg}(A, R_i, H_j)$$

- Finally, if R_i is positive and M_j is negative or viceversa,

$$diff_{bag-adapted}(A, R_i, M_j) = diff_{bag-max}(A, R_i, M_j)$$

Finally, the function $diff_{instance}$ applied to particular instances calculates the sum of the distances of all attributes (Manhattan distance, (10)). When dealing with nominal attributes, function $diff_{instance}(A, I_x, I_y)$ is defined as:

$$diff_{instance}(A, I_x, I_y) = \begin{cases} 0; & \text{value(A, } I_x\text{) = value(A, } I_y\text{)} \\ 1; & \text{otherwise} \end{cases}$$

and for numerical attributes as:

$$diff_{instance}(A, I_x, I_y) = \frac{|value(A, I_x) - value(A, I_y)|}{max(A) - min(A)}$$

where I_x is an instance x that belongs to a bag of the data set, I_y is another instance that belong to a bag in the data set, and A is the attribute in question.

3.2. Genetic Algorithm

Genetic algorithms (GAs) are best known for their ability to efficiently search large spaces about which little is known a priori (11). Since GAs are relatively insensitive to noise, they have proven to be an excellent choice for a basis of a more robust feature selection strategy and ever since the pioneering work by Siedlecki and Sklansky (38), many published studies have shown the advantages of the GAs in feature selection or learning in traditional supervised learning settings as individual models (24; 34; 28) as ensembles of the models (40).

3.2.1. Specification of the Algorithm

The algorithm used in this work is an elitist generational algorithm where a population of individuals is randomly generated at the beginning, and several genetic operations (crossover and mutation) are applied for a determined number of generations. The specification of individuals, genetic operators, and fitness functions follows.

In our algorithm, individuals are coded as vectors of binary genes, each individual containing one gene for each feature which can take 2 values, 0 meaning that the feature is not selected, and 1 meaning that the feature is selected. This representation allows us to consider the space of all possible feature subsets and is the most common way to code individuals in this field.

The crossover operation used is one point crossover. This crossover consists in slicing each parent chromosome at 1 random point, and replacing half of one chromosome with half of the second chromosome.

The mutation operation used is a locus mutation. This mutator consists in selecting all genes with a probability of changing a binary element to 1 or to 0.

Finally, the fitness value is a measure of the individual's performance in the problem at hand. For each individual, the result of the classification using the features that are selected by an individual (that is, the features corresponding to '1's in the chromosome) are computed and the product of *sensitivity* · *specificity* estimated by the classifier is returned as the fitness. As it is an wrapper method, the classifier in each case will be the specific classifier to be evaluated: that is, the feature selection is included as a phase of the classifier itself.

3.3. HyDR-MI Algorithm

Our approach combines both methods to achieve one precise and efficient method. Thus, the ReliefF-MI method developed (Section 3.1) is capable of handling continuous valued features, multiple target classes and noisy data. For each feature, ReliefF-MI returns a numerical measure of its importance and a threshold determines which features are irrelevant. This method's main problems are that many features in real domains have high correlations and thus many are (weakly) relevant; since this method cannot recognise them as irrelevant (or redundant), its results can be too general because they are independent of the method.

A practical and effective way of overcoming this limitation is to develop a hybrid algorithm which incorporates domain-specific knowledge. To overcome these faults, a GA (Section 3.2.1) is chosen to search through the space of features in the wrapper component of our hybrid. A GA is suited to this type of problem because it performs a randomized search and is not very susceptible to getting stuck in local minima. Moreover, the main problem of these subset selection methods is their computation time. So this combination with the filter method which directly eliminates all irrelevant features reduces the search space to achieve better results in less time.

Figure 3 shows the main steps for the proposed hybrid algorithm (HyDR-MI) based on feature relevance and using the filter and wrapper method.

4. Empirical Study

Our experimental study is aimed to demonstrate the effect of dimensionality reduction in MIL by means of the proposed feature selection hybrid. We estimate the efficacy of feature selection on a representative number of different types of classification algorithms. Particularly, the experimental study compares the performance of seventeen most popular classification approaches in MIL on a collection of five MIL benchmark datasets presenting drug activity prediction and content-based image categorization domains. In this section first, the experimental settings are described. Then, the experimental results are reported and discussed. The first experiment is used to evaluate the different ReliefF-MI techniques. Then, the technique showing the best performance is used as the filter component in the HyDR-MI method.

4.1. Experimental Setting

This section describes the application domains, the algorithms used to compare results, and the parameter settings of the feature selection models.

4.1.1. Problem Domains Used in Experiments

Our experiments consider two different problem domains, drug activity prediction and content-based image classification. Both applications are well-known in MIL. Table 1 summarizes detailed information about five MIL benchmark datasets used in the experiments to evaluate our hybrid method.

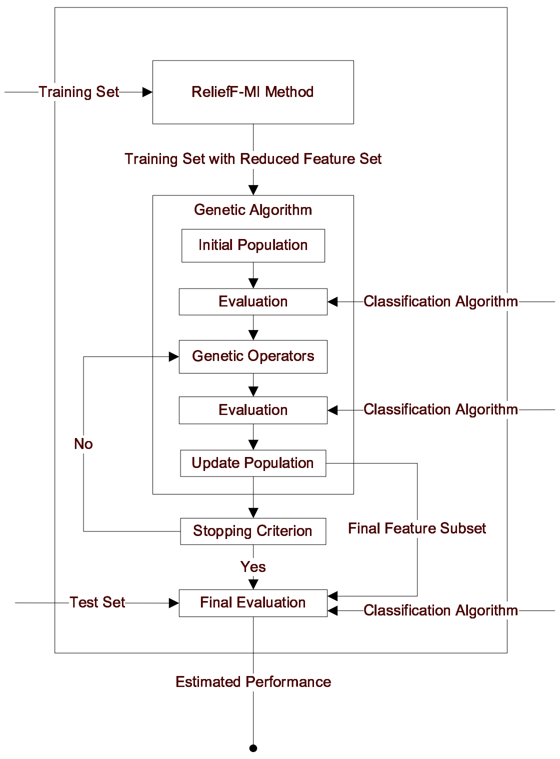


Figure 3: HyDR-MI Algorithm

- *The prediction of drug activity* was the first application of MIL. The problem consists of determining whether a drug molecule will bind strongly to a target protein. Each molecule may adopt a wide range of shapes or conformations. A positive molecule has at least one shape that can bind well (although it is not known which one) and a negative molecule means none of its shapes can make the molecule bind well. This problem could be represented in a very natural way in MIL settings: each molecule would be a bag and the conformations it can adopt would be the instances in that bag.

  The experiments presented here consider two data sets for the drug activity prediction problem (12), Musk1 and Musk2.

- *The content-based image classification* consists of identifying the intended target object(s) in images. The main difficulty of this problem is due to the fact that an image may contain multiple, possibly heterogeneous objects and we are interested in identifying only one specific target object(s). Thus, the global description of a whole image is too coarse to achieve good classification and retrieval, accuracy being a difficult problem in the supervised learning setting. From an MIL perspective, each image can be treated as a bag of segments which are modeled as instances, and concept points representing the target object can be learned through MIL algorithms. Content-based image classification has been a widely used method in MIL, there are numerous treatments in the literature, showing the advantage of solving this problem using multi-instance representation (30; 45; 46).

The experiments presented here consider three data sets representing categories of animals (2), specifically elephants, foxes and tigers. Each data set consists of 100 images which contain the animal and another 100 images which contain different animals using 230 features. The final goal consists in distinguishing some images of the animal in question from other images that do not contain it.

Table 1: General Information about Data Sets

Dataset	Bags			Attributes	Instances	Average
	Positive	*Negative*	*Total*			Bag Size
Musk1	47	45	92	166	476	5.17
Musk2	39	63	102	166	6598	64.69
Elephant	100	100	200	230	1391	6.96
Tiger	100	100	200	230	1220	6.10
Fox	100	100	200	230	1320	6.60

4.1.2. Methods Used in the Experiments

This study has considered some of the most representative paradigms used in MIL that have shown good results in different MIL applications. The main paradigms considered are methods based on diverse density, logistic regression, support vector machines, distance, decision tree, rules and naive Bayes. We discuss them briefly here. Interested readers are adviced to look into WEKA machine learning library for the implementation details and more in-depth information (43).

- *Methods based on Diverse Density.* Diverse Density (DD), proposed by Maron and Lozano-Perez (27), is perhaps the best known framework for MI learning. Given a set of positive and negative bags, the idea behind this approach is to learn a concept that is *close* to at least one instance in each positive bag, but *far* from all instances in all the negative bags. Thus, the concept must describe a region of instance space that is *dense* in instances from positive bags, and is also *diverse* in that it describes every positive bag. The experiments considers three algorithms based on classical diverse density algorithms for MI learning.
  - *MIDD* is an implementation of the standard DD algorithm that maximizes bag-level likelihood using the noisy-or model at training time. The class label of a new bag is assigned based on the class that receives the maximum probability.
  - *MIEMDD* is the expectation maximization diverse density (EMDD) algorithm (58). This algorithm combines the DD algorithm with an iterative approach inspired by the expectation maximization algorithm (EM). In each iteration, the most-likely-cause model (27) is used to find the most likely target points based on the DD model that has been learned.
- *Methods based on Logistic Regression.* Logistic Regression is a popular machine learning method in standard single instance learning (32). Our experiments consider a Multiple Instance Logistic Regression algorithm *MILR* which adapts standard SI logistic regression to

MI learning assuming a standard single instance logistic model at the instance level and using its class probabilities to compute bag-level class probabilities using the noisy-or model employed by DD. Because the instance-level class labels are not known, MILR learns the parameters of this MI logistic regression model by maximizing bag-level likelihood.

- *Distance-based Approaches.* The k-nearest neighbour (k-NN) in an MIL framework was introduced by Wang and Zucker (41). The main difference between the different approaches for using nearest neighbour algorithms lies in the definition of the distance metric used to measure the distance between bags. Two schemes widely employed are minimal Hausdorff distance and Kullback-Leibler distance. The experimentation considers:
  - *CitationKNN* which is a nearest-neighbour-based approach for the MI problems proposed by Wang and Zucker (41). It measures the distance between bags using the Hausdorff distance. In contrast to standard nearest-neighbour learning where only the nearest neighbours of an example to be classified are considered, the classification process of CitationKNN considers those examples in the training set for which the example to be classified is the nearest both in references and citers.
  - *MIOptimalBall* is based on the optimal ball method (3). It implements the idea of classification based on the distance to a reference point. More specifically, this method attempts to locate a *ball* in instance space so that all instances of all negative bags are outside the ball and at least one instance of each positive bag is inside the ball.
- *Methods based on rules:* these methods generate decision rules which can be understood by the user. They employ different classic algorithms of traditional supervised learning adapted to multiple instance learning. There are two possible ways to adapt them. The first is MIWrapper (43), a method that assigns the class label of a bag to all its instances and then trains a single instance algorithm on the resulting data, while the second one, MISimple (43), computes summary statistics for a bag to form a single instance. The algorithms considered in this paradigm are: PART(MIWrapper), PART(MISimple) and combinations of the different techniques using rule based systems: AdaBoost&PART(MISimple), Bagging&PART(MIWrapper) and AdaBoost&PART(MIWrapper).
- *Methods based on Support Vector Machines:* a support vector machine (SVM) is a popular recent development within the machine learning and data mining communities. This approach has a great number of adaptations to the MIL framework (2; 39; 58) whose results perform well in different applications. The experiments consider two algorithms: the MISMO algorithm which replaces the single-instance kernel function with a multi-instance kernel (i.e., an appropriate similarity function that is based on bags of instances), and the SMO algorithm (22) for SVM learning in conjunction with an MI kernel (17) that uses the procedure based on the MIWrapper mentioned previously for the adaptation.
- *Methods based on decision trees:* these methods are inspired by AdaBoost, which builds a series of weak classifiers using a single instance learner based on appropriately re-weighted versions of the input data where all instances receive their bags' labels (44). The algorithms considered in this paradigm are: DecisionStump (43) and RepTree (43).
- *Methods based on naive Bayes:* naive Bayes (NB) is a simple probabilistic classifier based on applying Bayes' theorem. This method is adapted to multiple instance learning using

the above-mentioned procedure based on the MIWrapper and using the Naive Bayes (43) algorithm.

4.1.3. Experimental Setup

This section describes the main characteristics of the experiment setup. We have conducted two series of experiments. First, an instantiation of ReliefF-MI (the filter component in our hybrid method) is evaluated with different feature metrics. Then HyDR-MI, which searchers for the best configuration of ReliefF-MI, is applied and evaluated to determine the best feature subset.

In all experiments we use the 10-fold cross validation method to estimate the generalization performance of the classifiers trained using a combination of selected features. Moreover, five different seeds are used on HyDR-MI method due to its stochasticity and the average results are shown. All information about datasets is available at *http://www.uco.es/grupos/kdis/mil/fs*.

Next, the parameters used in our technique are specified. This parameter setting was finally selected after several parameter tuning experiments. With respect to the configuration of ReliefF-MI, the 80 nearest neighbours ($k = 80$) are used and 180 iterations are carried out ($m = 180$). Finally, the threshold to delimit the number of features is limited to a fixed number of features where different percentages of features are used. With respect to GA specification, 50 generations are used with 100 individuals in each generation. The number is reduced to avoid an excessive computation time. The crossover and mutation rates are 90% and 30%, respectively. The initial population is generated randomly and elitism is used, considering the best individual from the previous generation.

4.2. Results and Discussion

Two types of experiments are carried out to evaluate the performance of HyDR-MI. In the first set of experiments, the proposed filter method is evaluated, considering different metrics. Then, HyDR-MI uses the best configuration of ReliefF-MI as the filter component and the second experiment evaluates and compares HyDR-MI to ReliefF-MI and the use of all features.

4.2.1. Experimental Results with ReliefF-MI

Experimental results with different algorithms and data sets are discussed and compared to show the efficiency of different versions of ReliefF-MI. Generally, the feature selection method based on filters requires a threshold to choose a specific feature subset. To show the effectiveness of the different methods, the study considers different sized feature sets. Thus, starting from the original feature set with all features, six different sets are generated. The first of them is composed of only 10% of the most relevant features, the next has 20% of the most relevant features and the others, 30%, 40%, 50% and 60% of the most relevant features. The output of each feature selection method is evaluated by 17 different algorithms and 5 data sets according to the use of the different percentages of features.

Comparing Different ReliefF-MI Metrics. Numerical results corresponding to the different setting of ReliefF-MI are reported in Table 2 (for minimal Hausdorff distance), Table 3 (for maximal Hausdorff distance), Table 4 (for average Hausdorff distance), and Table 5 (for adapted Hausdorff distance). With this information, it is very complicated to decide on the best metric, so a statistical test is used to evaluate the results of the different datasets and algorithms and compare the different distances, verifying if different metrics in ReliefF-MI are affected by different feature set sizes. The test selected is the Friedman test (16; 15), a non-parametric test that compares the

Table 2: Results using ReliefF-MI with minimal Hausdorff distance

Algorithms	10%					20%					30%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.850	0.630	0.745	0.722	0.730	0.845	0.615	0.760	0.744	0.720	0.800	0.595	0.780	0.778	0.720
MDD	0.800	0.600	0.710	0.811	0.720	0.760	0.655	0.800	0.767	0.750	0.770	0.690	0.795	0.800	0.800
MIBoost (RepTree)	0.845	0.665	0.840	0.733	0.720	0.850	0.665	0.850	0.733	0.880	0.900	0.685	0.825	0.789	0.830
MIBoost (DecisionStump)	0.785	0.695	0.820	0.822	0.770	0.790	0.695	0.810	0.711	0.890	0.780	0.645	0.815	0.700	0.810
MIDD	0.780	0.645	0.750	0.700	0.620	0.735	0.675	0.815	0.700	0.670	0.720	0.695	0.805	0.700	0.780
MIEMDD	0.720	0.605	0.685	0.822	0.760	0.755	0.665	0.735	0.822	0.770	0.715	0.590	0.765	0.811	0.820
MIRL	0.825	0.630	0.840	0.689	0.760	0.785	0.605	0.825	0.656	0.800	0.795	0.585	0.835	0.678	0.750
MIOptimalBall	0.715	0.495	0.765	0.778	0.650	0.725	0.500	0.750	0.733	0.650	0.670	0.525	0.740	0.733	0.690
MISMO (RBF Kernel)	0.865	0.655	0.800	0.689	0.770	0.785	0.620	0.795	0.700	0.850	0.795	0.640	0.820	0.700	0.840
MISMO (Polynomial Kernel)	0.825	0.685	0.780	0.633	0.700	0.840	0.635	0.795	0.778	0.810	0.820	0.615	0.795	0.678	0.840
MIWrapper (AdaBoost&PART)	0.825	0.745	0.830	0.711	0.810	0.825	0.660	0.855	0.800	0.850	0.810	0.660	0.805	0.800	0.880
MIWrapper (Bagging&PART)	0.865	0.595	0.810	0.800	0.810	0.870	0.605	0.835	0.822	0.870	0.825	0.600	0.855	0.856	0.880
MIWrapper (PART)	0.830	0.615	0.815	0.844	0.760	0.810	0.555	0.785	0.789	0.840	0.805	0.550	0.815	0.800	0.840
MIWrapper (SMO)	0.835	0.675	0.715	0.589	0.600	0.795	0.670	0.705	0.711	0.670	0.795	0.650	0.720	0.744	0.710
MIWrapper (Naive Bayes)	0.815	0.650	0.675	0.789	0.730	0.800	0.615	0.715	0.800	0.740	0.760	0.605	0.655	0.811	0.770
MISimple (AdaBoost&PART)	0.830	0.600	0.840	0.756	0.840	0.795	0.640	0.810	0.800	0.810	0.850	0.650	0.800	0.867	0.800
MISimple (PART)	0.730	0.670	0.765	0.800	0.800	0.795	0.650	0.785	0.811	0.800	0.785	0.615	0.750	0.733	0.810

Algorithms	40%					50%					60%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.780	0.615	0.785	0.856	0.740	0.500	0.500	0.500	0.856	0.780	0.500	0.500	0.500	0.867	0.780
MDD	0.765	0.690	0.785	0.856	0.780	0.745	0.705	0.780	0.856	0.790	0.735	0.710	0.790	0.856	0.800
MIBoost (RepTree)	0.840	0.670	0.815	0.844	0.820	0.825	0.670	0.815	0.833	0.830	0.825	0.670	0.815	0.822	0.830
MIBoost (DecisionStump)	0.780	0.650	0.815	0.700	0.830	0.780	0.650	0.815	0.722	0.820	0.780	0.650	0.815	0.689	0.810
MIDD	0.725	0.675	0.780	0.789	0.770	0.730	0.660	0.785	0.867	0.790	0.730	0.665	0.795	0.900	0.800
MIEMDD	0.770	0.650	0.740	0.844	0.820	0.755	0.585	0.765	0.911	0.750	0.765	0.650	0.740	0.933	0.800
MIRL	0.835	0.535	0.815	0.667	0.780	0.845	0.520	0.790	0.744	0.820	0.845	0.525	0.785	0.767	0.830
MIOptimalBall	0.630	0.520	0.705	0.722	0.740	0.630	0.530	0.730	0.711	0.760	0.620	0.530	0.730	0.656	0.730
MISMO (RBF Kernel)	0.785	0.610	0.835	0.778	0.870	0.795	0.585	0.800	0.778	0.860	0.790	0.585	0.800	0.744	0.870
MISMO (Polynomial Kernel)	0.795	0.615	0.790	0.856	0.840	0.790	0.580	0.790	0.844	0.840	0.795	0.585	0.790	0.833	0.860
MIWrapper (AdaBoost&PART)	0.820	0.685	0.870	0.822	0.870	0.790	0.685	0.840	0.911	0.860	0.790	0.685	0.840	0.867	0.900
MIWrapper (Bagging&PART)	0.815	0.615	0.855	0.889	0.840	0.810	0.600	0.845	0.878	0.850	0.810	0.600	0.845	0.922	0.840
MIWrapper (PART)	0.760	0.545	0.785	0.800	0.840	0.780	0.550	0.790	0.811	0.830	0.780	0.550	0.790	0.800	0.830
MIWrapper (SMO)	0.785	0.645	0.710	0.767	0.720	0.800	0.635	0.715	0.767	0.730	0.800	0.635	0.715	0.833	0.760
MIWrapper (Naive Bayes)	0.770	0.590	0.670	0.811	0.740	0.760	0.590	0.680	0.833	0.660	0.760	0.590	0.680	0.844	0.690
MISimple (AdaBoost&PART)	0.810	0.655	0.820	0.833	0.830	0.805	0.630	0.845	0.889	0.830	0.795	0.630	0.840	0.789	0.770
MISimple (PART)	0.755	0.625	0.780	0.778	0.790	0.765	0.635	0.765	0.789	0.830	0.765	0.635	0.765	0.767	0.770

Table 3: Results using ReliefF-MI with maximal Hausdorff distance

Algorithms	10%					20%					30%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.830	0.615	0.750	0.756	0.660	0.825	0.650	0.730	0.822	0.690	0.805	0.595	0.765	0.811	0.800
MDD	0.810	0.620	0.725	0.744	0.660	0.735	0.685	0.775	0.756	0.720	0.760	0.680	0.800	0.744	0.760
MIBoost (RepTree)	0.870	0.655	0.825	0.811	0.790	0.880	0.710	0.855	0.867	0.820	0.830	0.680	0.810	0.778	0.820
MIBoost (DecisionStump)	0.800	0.655	0.825	0.700	0.780	0.805	0.650	0.805	0.778	0.800	0.780	0.635	0.815	0.700	0.800
MIDD	0.780	0.600	0.755	0.778	0.570	0.755	0.660	0.765	0.811	0.610	0.740	0.685	0.795	0.800	0.670
MIEMDD	0.775	0.530	0.725	0.856	0.700	0.730	0.605	0.770	0.811	0.790	0.715	0.590	0.750	0.800	0.800
MIRL	0.855	0.600	0.815	0.733	0.740	0.795	0.580	0.775	0.700	0.730	0.815	0.585	0.815	0.767	0.790
MIOptimalBall	0.740	0.575	0.795	0.678	0.640	0.720	0.555	0.760	0.700	0.720	0.665	0.530	0.715	0.733	0.670
MISMO (RBF Kernel)	0.835	0.615	0.765	0.756	0.600	0.810	0.620	0.775	0.700	0.790	0.790	0.610	0.805	0.722	0.840
MISMO (Polynomial Kernel)	0.825	0.620	0.765	0.578	0.800	0.830	0.650	0.790	0.722	0.860	0.815	0.610	0.790	0.822	0.920
MIWrapper (AdaBoost&PART)	0.840	0.615	0.830	0.811	0.780	0.850	0.655	0.850	0.811	0.840	0.855	0.615	0.850	0.800	0.840
MIWrapper (Bagging&PART)	0.850	0.585	0.830	0.722	0.780	0.855	0.620	0.860	0.767	0.810	0.825	0.600	0.830	0.789	0.830
MIWrapper (PART)	0.815	0.580	0.830	0.722	0.770	0.795	0.550	0.795	0.789	0.780	0.800	0.550	0.770	0.789	0.810
MIWrapper (SMO)	0.815	0.660	0.705	0.689	0.600	0.810	0.645	0.695	0.733	0.640	0.810	0.650	0.710	0.756	0.690
MIWrapper (Naive Bayes)	0.820	0.590	0.655	0.656	0.680	0.770	0.620	0.670	0.667	0.720	0.715	0.605	0.670	0.667	0.720
MISimple (AdaBoost&PART)	0.855	0.570	0.800	0.811	0.760	0.825	0.675	0.825	0.789	0.810	0.840	0.625	0.845	0.800	0.820
MISimple (PART)	0.795	0.620	0.770	0.700	0.670	0.730	0.625	0.740	0.700	0.770	0.785	0.635	0.775	0.756	0.770

Algorithms	40%					50%					60%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.770	0.600	0.775	0.878	0.760	0.775	0.620	0.500	0.867	0.780	0.500	0.500	0.500	0.844	0.770
MDD	0.740	0.680	0.780	0.811	0.710	0.755	0.690	0.770	0.844	0.750	0.760	0.705	0.785	0.833	0.770
MIBoost (RepTree)	0.820	0.650	0.815	0.878	0.830	0.825	0.670	0.815	0.867	0.830	0.825	0.670	0.815	0.867	0.800
MIBoost (DecisionStump)	0.780	0.650	0.815	0.711	0.840	0.780	0.650	0.815	0.711	0.830	0.780	0.650	0.815	0.711	0.810
MIDD	0.715	0.690	0.805	0.844	0.750	0.745	0.630	0.780	0.844	0.790	0.735	0.660	0.795	0.811	0.770
MIEMDD	0.745	0.615	0.775	0.878	0.800	0.760	0.620	0.765	0.833	0.800	0.775	0.625	0.745	0.844	0.760
MIRL	0.855	0.535	0.790	0.767	0.760	0.855	0.520	0.790	0.689	0.810	0.840	0.510	0.790	0.756	0.800
MIOptimalBall	0.665	0.525	0.740	0.778	0.730	0.625	0.530	0.730	0.700	0.650	0.625	0.530	0.730	0.711	0.690
MISMO (RBF Kernel)	0.805	0.590	0.815	0.744	0.820	0.795	0.590	0.800	0.756	0.810	0.795	0.590	0.800	0.722	0.860
MISMO (Polynomial Kernel)	0.790	0.585	0.795	0.867	0.890	0.785	0.595	0.790	0.878	0.890	0.785	0.595	0.790	0.900	0.900
MIWrapper (AdaBoost&PART)	0.845	0.660	0.840	0.867	0.860	0.810	0.670	0.840	0.844	0.860	0.790	0.685	0.840	0.878	0.870
MIWrapper (Bagging&PART)	0.830	0.590	0.830	0.844	0.850	0.815	0.610	0.845	0.867	0.860	0.810	0.600	0.845	0.856	0.880
MIWrapper (PART)	0.760	0.550	0.785	0.789	0.830	0.795	0.550	0.790	0.833	0.830	0.780	0.550	0.790	0.800	0.850
MIWrapper (SMO)	0.800	0.645	0.710	0.811	0.720	0.805	0.635	0.715	0.822	0.720	0.800	0.635	0.715	0.833	0.740
MIWrapper (Naive Bayes)	0.735	0.590	0.690	0.667	0.750	0.755	0.595	0.680	0.689	0.770	0.760	0.590	0.680	0.711	0.760
MISimple (AdaBoost&PART)	0.840	0.655	0.850	0.811	0.820	0.795	0.620	0.840	0.856	0.870	0.795	0.625	0.840	0.889	0.880
MISimple (PART)	0.785	0.615	0.770	0.856	0.820	0.760	0.635	0.765	0.789	0.810	0.765	0.635	0.765	0.811	0.830

Table 4: Results using ReliefF-MI with average Hausdorff distance

Algorithms	10%					20%					30%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.840	0.610	0.745	0.778	0.700	0.830	0.630	0.740	0.756	0.740	0.800	0.580	0.780	0.822	0.740
MDD	0.790	0.605	0.710	0.789	0.710	0.750	0.675	0.790	0.778	0.750	0.750	0.715	0.800	0.767	0.700
MIBoost (RepTree)	0.865	0.700	0.840	0.778	0.810	0.850	0.660	0.840	0.789	0.830	0.860	0.675	0.820	0.778	0.840
MIBoost (DecisionStump)	0.800	0.660	0.820	0.778	0.810	0.795	0.660	0.815	0.700	0.830	0.780	0.660	0.815	0.767	0.820
MIDD	0.780	0.595	0.750	0.689	0.520	0.760	0.685	0.810	0.733	0.710	0.730	0.625	0.805	0.733	0.730
MIEMDD	0.745	0.530	0.685	0.822	0.650	0.740	0.655	0.735	0.833	0.770	0.720	0.625	0.775	0.856	0.810
MIRL	0.840	0.615	0.840	0.656	0.770	0.795	0.590	0.750	0.711	0.720	0.800	0.580	0.850	0.844	0.770
MIOptimalBall	0.735	0.525	0.765	0.756	0.620	0.730	0.520	0.745	0.689	0.670	0.675	0.520	0.730	0.722	0.650
MISMO (RBF Kernel)	0.830	0.655	0.800	0.656	0.650	0.785	0.615	0.775	0.711	0.760	0.795	0.625	0.835	0.767	0.860
MISMO (Polynomial Kernel)	0.830	0.665	0.780	0.644	0.700	0.825	0.610	0.780	0.733	0.830	0.815	0.630	0.805	0.600	0.890
MIWrapper (AdaBoost&PART)	0.820	0.620	0.830	0.644	0.750	0.845	0.605	0.860	0.756	0.830	0.840	0.670	0.840	0.856	0.870
MIWrapper (Bagging&PART)	0.860	0.610	0.810	0.744	0.840	0.875	0.580	0.845	0.789	0.870	0.820	0.585	0.855	0.844	0.850
MIWrapper (PART)	0.810	0.570	0.815	0.789	0.790	0.800	0.570	0.840	0.767	0.830	0.810	0.580	0.785	0.767	0.830
MIWrapper (SMO)	0.830	0.655	0.715	0.656	0.520	0.805	0.680	0.705	0.667	0.690	0.810	0.650	0.715	0.811	0.730
MIWrapper (Naive Bayes)	0.825	0.585	0.675	0.767	0.660	0.775	0.605	0.685	0.767	0.660	0.730	0.600	0.675	0.833	0.690
MISimple (AdaBoost&PART)	0.840	0.560	0.840	0.789	0.740	0.810	0.665	0.820	0.800	0.810	0.830	0.605	0.820	0.844	0.860
MISimple (PART)	0.740	0.660	0.765	0.733	0.710	0.765	0.705	0.770	0.767	0.740	0.780	0.605	0.725	0.730	0.770

Algorithms	40%					50%					60%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.785	0.615	0.785	0.878	0.780	0.780	0.615	0.500	0.900	0.840	0.500	0.500	0.500	0.911	0.800
MDD	0.755	0.705	0.785	0.811	0.710	0.730	0.710	0.780	0.833	0.710	0.750	0.710	0.795	0.867	0.740
MIBoost (RepTree)	0.825	0.665	0.815	0.844	0.870	0.825	0.665	0.815	0.844	0.840	0.825	0.670	0.815	0.867	0.840
MIBoost (DecisionStump)	0.780	0.655	0.815	0.722	0.840	0.780	0.650	0.815	0.711	0.820	0.780	0.650	0.815	0.711	0.840
MIDD	0.740	0.655	0.790	0.744	0.770	0.735	0.660	0.780	0.756	0.770	0.745	0.650	0.790	0.844	0.760
MIEMDD	0.745	0.670	0.760	0.844	0.810	0.775	0.615	0.765	0.822	0.810	0.775	0.625	0.755	0.867	0.750
MIRL	0.835	0.525	0.805	0.733	0.800	0.845	0.490	0.785	0.733	0.830	0.840	0.510	0.790	0.700	0.840
MIOptimalBall	0.630	0.520	0.710	0.689	0.700	0.620	0.530	0.730	0.678	0.740	0.625	0.530	0.730	0.700	0.720
MISMO (RBF Kernel)	0.800	0.615	0.845	0.767	0.870	0.800	0.605	0.800	0.733	0.890	0.795	0.590	0.800	0.711	0.870
MISMO (Polynomial Kernel)	0.795	0.600	0.790	0.844	0.880	0.790	0.585	0.790	0.856	0.900	0.785	0.580	0.790	0.856	0.900
MIWrapper (AdaBoost&PART)	0.830	0.665	0.845	0.889	0.900	0.795	0.665	0.840	0.889	0.910	0.790	0.685	0.840	0.867	0.890
MIWrapper (Bagging&PART)	0.810	0.600	0.850	0.867	0.850	0.810	0.600	0.845	0.889	0.830	0.810	0.600	0.845	0.900	0.810
MIWrapper (PART)	0.765	0.540	0.780	0.833	0.840	0.790	0.550	0.790	0.833	0.830	0.780	0.550	0.790	0.822	0.850
MIWrapper (SMO)	0.795	0.640	0.710	0.833	0.720	0.795	0.635	0.715	0.833	0.740	0.800	0.635	0.715	0.833	0.740
MIWrapper (Naive Bayes)	0.750	0.595	0.660	0.811	0.700	0.760	0.590	0.680	0.811	0.700	0.760	0.590	0.680	0.844	0.730
MISimple (AdaBoost&PART)	0.815	0.655	0.840	0.767	0.840	0.790	0.645	0.840	0.856	0.860	0.795	0.625	0.840	0.778	0.790
MISimple (PART)	0.765	0.635	0.775	0.722	0.800	0.765	0.635	0.765	0.778	0.780	0.765	0.635	0.765	0.711	0.800

Table 5: Results using ReliefF-MI with adapted Hausdorff distance

Algorithms	10%					20%					30%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.815	0.615	0.745	0.867	0.760	0.830	0.630	0.745	0.889	0.800	0.805	0.570	0.755	0.844	0.800
MDD	0.805	0.660	0.705	0.744	0.630	0.745	0.680	0.765	0.744	0.670	0.745	0.655	0.800	0.767	0.720
MIBoost (RepTree)	0.855	0.710	0.840	0.744	0.840	0.845	0.655	0.855	0.867	0.860	0.820	0.685	0.825	0.844	0.840
MIBoost (DecisionStump)	0.805	0.700	0.830	0.700	0.770	0.805	0.660	0.805	0.711	0.850	0.780	0.670	0.815	0.722	0.820
MIDD	0.770	0.695	0.755	0.800	0.610	0.740	0.660	0.790	0.756	0.640	0.740	0.670	0.805	0.778	0.670
MIEMDD	0.770	0.615	0.715	0.867	0.780	0.700	0.635	0.735	0.800	0.710	0.730	0.660	0.760	0.856	0.750
MIRL	0.875	0.635	0.835	0.678	0.770	0.780	0.605	0.810	0.744	0.760	0.830	0.575	0.815	0.756	0.790
MIOptimalBall	0.740	0.535	0.775	0.733	0.610	0.720	0.540	0.720	0.711	0.610	0.665	0.520	0.745	0.711	0.620
MISMO (RBF Kernel)	0.855	0.650	0.785	0.656	0.610	0.795	0.620	0.785	0.700	0.760	0.795	0.600	0.830	0.700	0.840
MISMO (Polynomial Kernel)	0.820	0.655	0.770	0.611	0.610	0.815	0.645	0.790	0.667	0.760	0.815	0.635	0.785	0.744	0.840
MIWrapper (AdaBoost&PART)	0.860	0.665	0.840	0.867	0.820	0.825	0.675	0.850	0.833	0.830	0.840	0.645	0.845	0.867	0.860
MIWrapper (Bagging&PART)	0.865	0.605	0.830	0.767	0.810	0.850	0.615	0.835	0.822	0.860	0.860	0.605	0.840	0.811	0.830
MIWrapper (PART)	0.840	0.620	0.835	0.756	0.770	0.820	0.610	0.795	0.767	0.850	0.795	0.585	0.780	0.811	0.820
MIWrapper (SMO)	0.820	0.690	0.705	0.667	0.560	0.805	0.685	0.685	0.722	0.660	0.805	0.655	0.720	0.711	0.730
MIWrapper (Naive Bayes)	0.820	0.680	0.660	0.644	0.660	0.770	0.625	0.745	0.633	0.700	0.710	0.600	0.725	0.633	0.730
MISimple (AdaBoost&PART)	0.845	0.650	0.830	0.800	0.750	0.835	0.680	0.815	0.767	0.800	0.840	0.650	0.805	0.800	0.770
MISimple (PART)	0.780	0.665	0.775	0.722	0.710	0.740	0.660	0.760	0.700	0.720	0.780	0.670	0.780	0.767	0.730

Algorithms	40%					50%					60%				
	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2	Tiger	Fox	Elep	Musk1	Musk2
citationKNN	0.775	0.605	0.775	0.867	0.780	0.775	0.535	0.500	0.889	0.750	0.500	0.500	0.500	0.867	0.760
MDD	0.735	0.710	0.770	0.722	0.740	0.750	0.685	0.780	0.811	0.790	0.735	0.705	0.795	0.844	0.750
MIBoost (RepTree)	0.840	0.660	0.815	0.878	0.790	0.825	0.670	0.815	0.844	0.800	0.825	0.670	0.815	0.822	0.800
MIBoost (DecisionStump)	0.780	0.655	0.815	0.689	0.800	0.780	0.650	0.815	0.689	0.790	0.780	0.650	0.815	0.667	0.820
MIDD	0.700	0.665	0.820	0.778	0.750	0.755	0.665	0.785	0.867	0.790	0.750	0.675	0.790	0.911	0.790
MIEMDD	0.740	0.650	0.760	0.789	0.770	0.750	0.585	0.765	0.811	0.790	0.740	0.640	0.755	0.867	0.780
MIRL	0.850	0.545	0.825	0.756	0.800	0.850	0.510	0.790	0.722	0.800	0.840	0.515	0.790	0.744	0.800
MIOptimalBall	0.620	0.515	0.740	0.767	0.670	0.625	0.530	0.730	0.756	0.650	0.625	0.530	0.730	0.789	0.660
MISMO (RBF Kernel)	0.800	0.600	0.830	0.811	0.860	0.800	0.595	0.800	0.744	0.870	0.795	0.595	0.800	0.744	0.860
MISMO (Polynomial Kernel)	0.800	0.605	0.790	0.822	0.850	0.785	0.595	0.790	0.867	0.870	0.780	0.580	0.790	0.911	0.860
MIWrapper (AdaBoost&PART)	0.845	0.655	0.860	0.889	0.850	0.820	0.685	0.840	0.867	0.880	0.790	0.685	0.840	0.867	0.920
MIWrapper (Bagging&PART)	0.830	0.600	0.845	0.867	0.840	0.815	0.605	0.845	0.833	0.850	0.810	0.605	0.845	0.911	0.860
MIWrapper (PART)	0.775	0.540	0.820	0.778	0.820	0.790	0.540	0.790	0.778	0.810	0.780	0.550	0.790	0.800	0.870
MIWrapper (SMO)	0.795	0.630	0.710	0.811	0.710	0.800	0.635	0.715	0.811	0.710	0.800	0.635	0.715	0.844	0.720
MIWrapper (Naive Bayes)	0.730	0.585	0.700	0.656	0.710	0.760	0.590	0.680	0.689	0.730	0.760	0.590	0.680	0.767	0.770
MISimple (AdaBoost&PART)	0.835	0.615	0.835	0.856	0.830	0.805	0.635	0.840	0.867	0.830	0.795	0.635	0.840	0.867	0.840
MISimple (PART)	0.780	0.630	0.780	0.800	0.720	0.760	0.635	0.765	0.822	0.760	0.765	0.635	0.765	0.822	0.800

Table 6: Friedman Tests (comparison between metrics of ReliefF-MI)

Friedman Test			
Percentage of Features used	χ^2 (α = 0.95)	Value Test	Conclusion
10%	7.815	10.091	Reject null hypothesis
20%	7.815	5.636	Accept null hypothesis
30%	7.815	5.506	Accept null hypothesis
40%	7.815	3.822	Accept null hypothesis
50%	7.815	1.172	Accept null hypothesis
60%	7.815	0.618	Accept null hypothesis

average ranks of the methods considered, where the metric that achieves the highest accuracy for one percentage of features is given a rank of 1, the metric with the next highest accuracy value has the rank of 2, and so on. In this way, a metric's having the value closest to 1 indicates that the algorithms considered in this study generally obtain the best results using that particular percentage of features rather than any other percentage of features.

Table 6 reports the Friedman test results, which indicate that there are only significant differences between metrics when the method uses 10% of the most relevant features because, in this case, the null hypothesis is rejected. We decided to perform a post-hoc test, the Bonferroni-Dunn test (13), to find out what significant differences occurred when 10% of the features are used. Results of this test (Figure 4) show that the new metric designed in this paper is the best option because it has the lowest ranking value; the average Hausdorff distance and maximum Hausdorff distance produce significantly worse results because their ranking value is higher than the threshold set for this test (considering a 95% of confidence level, this value is set to 2.679 and it is represented in the figure by a horizontal line in bold). The problem of the other metrics could be for the following reason. The maximal Hausdorff distance calculates the minimum value so that each point in a set (set A or set B) can find at least one point in the other set (set B or set A). However, the two max notations in the definition above tend to determine the distance between the two sets based on outlier data points. Thus, there is an outlier data point in this metric that is far away from the other points and tends to dominate the distance result, thus modifying it. The average Hausdorff distance is also a problem when calculating the distance between the bags in a positive class, because these cases use an average distance of all the instances in each bag and

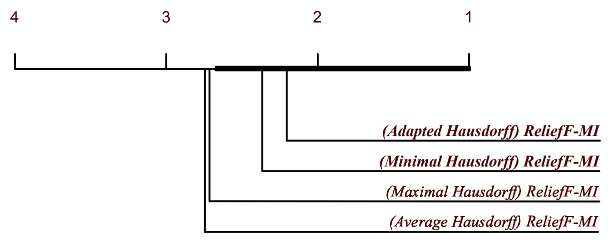


Figure 4: Bonferroni Dunn Test (p < 0.05) (comparison between metrics of ReliefF-MI)

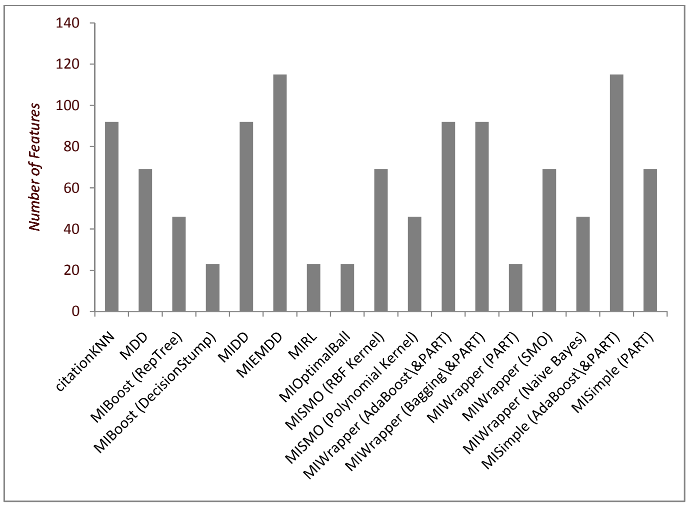


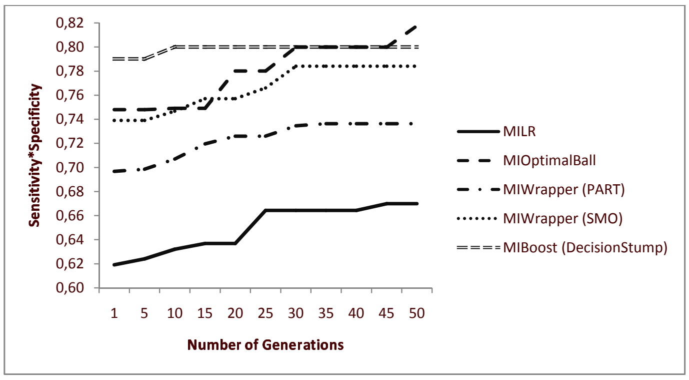
Figure 5: The best feature subset size with ReliefF-MI and the best results obtained by the different methods

there is no guarantee that all them are really positive instances (according to Dietterich et al.'s hypothesis, only it is known that at least one of them is positive). The new metric tries to solve this problem considering different scenarios as a function of the information in each bag.

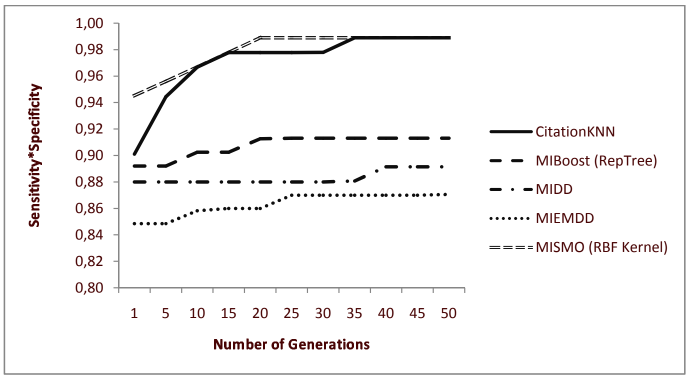
Study of the Weaknesses of ReliefF-MI. ReliefF-MI is an efficient method that sets a numerical value for each feature related to its significance. This method acts as an independent preprocessing method. Its most relevant shortcoming is that it delimits relevant and weakly relevant features (that is, determines that the final feature subset for algorithms achieves the best results for those features with a numerical value over 0). Figure 5 shows the most reduced number of features for each algorithm to obtain the best results. The results are obtained from a previous table considering the elephant dataset (a similar study done on the other datasets obtained similar results). At first glance, each algorithm seems to obtain the best results with a different number of features (the different percentages that have been shown in the previous section). This leads to the reflection that the optimal feature subset depends on the algorithm in question. Thus it is a complex task to set a general threshold for each test without taking into account the classifier. These results show that it would be advisable to consider a specific algorithm to set the final feature subset (wrapper methods). However, working with all features in these methods requires too much computational time. Thus, combining with a previous method (such as ReliefF-MI) which reduces the initially irrelevant features would seem a good way to facilitate the task of the wra“er methods.

4.2.2. Experimental Results with HyDR-MI

In this section, the experiments are mainly meant to show the effectiveness of HyDR-MI in finding the best feature subset for the different algorithms used. Experiments employ classification accuracy criteria and compare the HyDR-MI method to the ReliefF-MI method and to the use of the full feature set.



(a) Classification accuracy performance by iteration and best individual for several methods

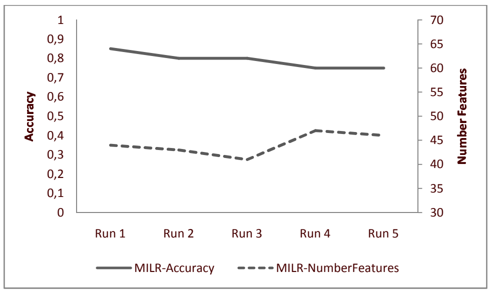


(b) Classification accuracy performance by iteration and best individual for several methods

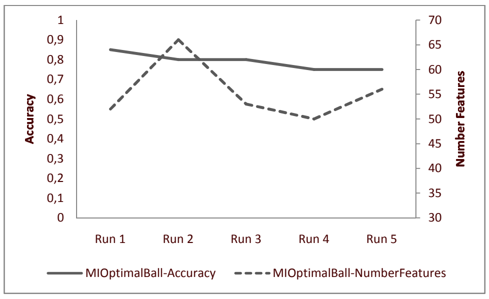
Figure 6: Runtime behaviour of HyDR-MI for a particular run of the elephant dataset

Before comparing results, we are going to consider the performance of the HyDR-MI algorithm. Figures 6(a) and 6(b) report on the behaviour in a particular run of the HyDR-MI algorithm for the elephant dataset using different algorithms in the various generations carried out. The best individual obtained for each generation is shown. Note that there were similar results for other datasets and algorithms, although they are not included here for the sake of brevity. The plots clearly indicate that there is monotone performance improvement during the search (the value shown is *sensitivity*·*specificity* which is the fitness function used to optimize the search for the best subset). Although some algorithms (such as the MIOptimalBall) might obtain better results using more generations because the convergence has still not been reached, other algorithms (such as MIBoost with DecisionStump) converge from the tenth iteration. However, most methods with this fixed number of generations obtain appropriate convergence.

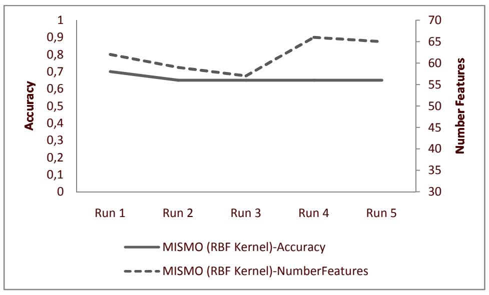
Next will be given the results for the different runs of each algorithm. Figure 7 plots five different runs of four algorithms for the elephant dataset and partition 1. These results are sorted by accuracy from the highest accuracy values to the lowest ones. This information shows that the results of the different runs are similar (there is not a great variance in the results). On the other hand the number of features in the different algorithms range from 30 to 65 features. Thus, some



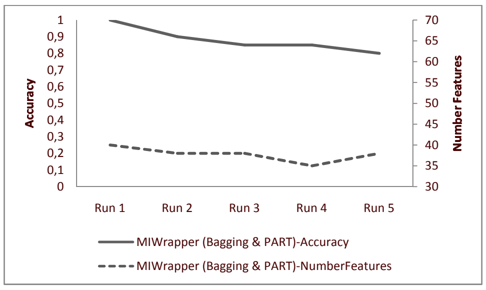
(a) Results of MILR algorithm



(b) Results of MIOptimalBall algorithm



(c) Results of MISMO (RBF Kernel) algorithm



(d) Results of MIWrapper (Bagging & PART) algorithm

Figure 7: Variation of classification accuracy as a function of the number of features selected for the different runs carried out on each algorithm in the elephant dataset

methods obtain better results with a higher number of features than others. For example, the MISMO algorithm (Figure 7(c)) obtains the best results using between 57 and 65 features while, on the other hand, the MIWrapper algorithm (Figure 7(d)) works better with a lower number, ranging between 35 and 40 features.

Tables 7, 8, and 9 show the accuracy results obtained by the different methods evaluating, respectively, the elephant, tiger, and fox datasets and Tables 10 and 11 show the results with musk1 and musk2 datasets. These tables show the accuracy values obtained by 17 different classifiers: when they incorporate the HyDR-MI algorithm; or when they use the feature subset provided by ReliefF-MI (the threshold used in these results is the same as the filter component in HyDR-MI, which consists of selecting 50% of the most relevant features for image categorization and 70% of the most relevant features for drug activity prediction, which then allows us to compare the difference between using only the filter component or the hybrid method); and finally when they use all the features. The best results for each algorithm and dataset is in bold. At first glance, it seems that for nearly all cases, HyDR-MI yields the best results in most classifiers except for a couple of methods that obtain better results with the other techniques. These results are relevant and show the advantages of using HyDR-MI from among the great variety of machine learning methods. Studying the information in greater detail, it can be seen that the algorithms differ in the amount of emphasis they place on feature selection. In some cases accuracy in future classification is significantly improved; in others, results are better, but the improvement is limited. Thus, at one extreme there are algorithms such as the simple nearest neighbour learner, that classifies

Table 7: Experimental results for general comparison (elephant dataset)

Algorithms	HyDR-MI		ReliefF-MI		Full Features	
	Acc	Features	Acc	Features	Acc	Features
citationKNN	**0.814**	**54**	0.500	115	0.500	230
MDD	0.787	52	0.780	115	**0.800**	**230**
MIBoost (RepTree)	**0.869**	**64**	0.815	115	0.815	230
MIBoost (DecisionStump)	**0.839**	**46**	0.815	115	0.815	230
MIDD	0.800	62	0.785	115	**0.825**	**230**
MIEMDD	**0.800**	**69**	0.765	115	0.730	230
MIRL	**0.822**	**43**	0.790	115	0.780	230
MIOptimalBall	**0.808**	**52**	0.730	115	0.730	230
MISMO (RBF Kernel)	**0.833**	**62**	0.800	115	0.800	230
MISMO (Polynomial Kernel)	**0.825**	**49**	0.790	115	0.790	230
MIWrapper (AdaBoost&PART)	**0.854**	**30**	0.840	115	0.840	230
MIWrapper (Bagging&PART)	**0.852**	**38**	0.845	115	0.845	230
MIWrapper (PART)	**0.822**	**64**	0.790	115	0.790	230
MIWrapper (SMO)	**0.819**	**61**	0.715	115	0.715	230
MIWrapper (Naive Bayes)	**0.843**	**62**	0.680	115	0.680	230
MISimple (AdaBoost&PART)	**0.813**	**32**	0.840	115	0.840	230
MISimple (PART)	**0.797**	**39**	0.765	115	0.765	230

Table 8: Experimental results for general comparison (tiger dataset)

Algorithms	HyDR-MI		ReliefF-MI		Full Features	
	Acc	Features	Acc	Features	Acc	Features
citationKNN	**0.834**	**61**	0.775	115	0.500	230
MDD	**0.758**	**55**	0.750	115	0.755	230
MIBoost (RepTree)	**0.850**	**40**	0.825	115	0.825	230
MIBoost (DecisionStump)	**0.862**	**30**	0.780	115	0.780	230
MIDD	**0.783**	**60**	0.755	115	0.740	230
MIEMDD	**0.807**	**53**	0.750	115	0.745	230
MIRL	0.817	46	**0.850**	**115**	0.840	230
MIOptimalBall	**0.809**	**49**	0.625	115	0.625	230
MISMO (RBF Kernel)	**0.832**	**61**	0.800	115	0.795	230
MISMO (Polynomial Kernel)	**0.839**	**59**	0.785	115	0.785	230
MIWrapper (AdaBoost&PART)	**0.859**	**33**	0.820	115	0.790	230
MIWrapper (Bagging&PART)	**0.859**	**41**	0.815	115	0.810	230
MIWrapper (PART)	**0.827**	**47**	0.790	115	0.780	230
MIWrapper (SMO)	**0.839**	**55**	0.800	115	0.800	230
MIWrapper (Naive Bayes)	**0.859**	**60**	0.760	115	0.760	230
MISimple (AdaBoost&PART)	**0.827**	**32**	0.805	115	0.795	230
MISimple (PART)	**0.814**	**43**	0.760	115	0.765	230

Table 9: Experimental results for general comparison (fox dataset)

Algorithms	HyDR-MI		ReliefF-MI		Full Features	
	Acc	Features	Acc	Features	Acc	Features
citationKNN	**0.648**	**59**	0.535	115	0.500	230
MDD	0.690	73	0.685	115	**0.700**	**230**
MIBoost (RepTree)	**0.685**	**52**	0.670	115	0.670	230
MIBoost (DecisionStump)	**0.709**	**31**	0.650	115	0.650	230
MIDD	**0.668**	**62**	0.665	115	0.655	230
MIEMDD	**0.682**	**55**	0.585	115	0.600	230
MIRL	**0.636**	**76**	0.510	115	0.510	230
MIOptimalBall	**0.629**	**43**	0.530	115	0.530	230
MISMO (RBF Kernel)	**0.651**	**61**	0.595	115	0.590	230
MISMO (Polynomial Kernel)	**0.634**	**62**	0.595	115	0.580	230
MIWrapper (AdaBoost&PART)	**0.715**	**39**	0.685	115	0.685	230
MIWrapper (Bagging&PART)	**0.635**	**64**	0.605	115	0.600	230
MIWrapper (PART)	**0.622**	**62**	0.540	115	0.550	230
MIWrapper (SMO)	**0.689**	**65**	0.635	115	0.635	230
MIWrapper (Naive Bayes)	**0.633**	**52**	0.590	115	0.590	230
MISimple (AdaBoost&PART)	**0.688**	**32**	0.635	115	0.625	230
MISimple (PART)	**0.691**	**66**	0.635	115	0.635	230

Table 10: Experimental results for general comparison (musk1 dataset)

Algorithms	HyDR-MI		ReliefF-MI		Full Features	
	Acc	Features	Acc	Features	Acc	Features
citationKNN	0.927	41	0.889	83	**0.944**	**166**
MDD	**0.813**	**46**	0.811	83	0.789	166
MIBoost (RepTree)	**0.876**	**31**	0.844	83	0.867	166
MIBoost (DecisionStump)	**0.809**	**47**	0.689	83	0.678	166
MIDD	0.911	45	0.867	83	**0.922**	**166**
MIEMDD	**0.848**	**57**	0.811	83	0.889	166
MIRL	**0.800**	**26**	0.722	83	0.733	166
MIOptimalBall	0.827	**56**	0.756	83	0.767	166
MISMO (RBF Kernel)	**0.793**	**52**	0.744	83	0.744	166
MISMO (Polynomial Kernel)	**0.911**	**53**	0.867	83	0.878	166
MIWrapper (AdaBoost&PART)	**0.907**	**29**	0.867	83	0.867	166
MIWrapper (Bagging&PART)	**0.927**	**28**	0.833	83	0.900	166
MIWrapper (PART)	**0.858**	**56**	0.778	83	0.800	166
MIWrapper (SMO)	**0.849**	**59**	0.811	83	0.844	166
MIWrapper (Naive Bayes)	**0.920**	**60**	0.689	83	0.789	166
MISimple (AdaBoost&PART)	**0.907**	**29**	0.867	83	0.756	166
MISimple (PART)	**0.876**	**37**	0.822	83	0.744	166

Table 11: Experimental results for general comparison (musk2 dataset)

Algorithms	HyDR-MI		ReliefF-MI		Full Features	
	Acc	Features	Acc	Features	Acc	Features
citationKNN	0.840	48	0.750	83	**0.850**	**166**
MDD	0.740	67	**0.790**	**83**	0.760	166
MIBoost (RepTree)	**0.900**	**41**	0.800	83	0.840	166
MIBoost (DecisionStump)	**0.920**	**41**	0.790	83	0.830	166
MIDD	0.750	67	**0.790**	**83**	0.730	166
MIEMDD	0.860	58	0.790	83	**0.900**	**166**
MIRL	**0.970**	**41**	0.800	83	0.840	166
MIOptimalBall	**0.850**	**58**	0.650	83	0.780	166
MISMO (RBF Kernel)	**0.910**	**50**	0.870	83	0.840	166
MISMO (Polynomial Kernel)	**0.920**	**55**	0.870	83	0.840	166
MIWrapper (AdaBoost&PART)	**0.920**	**41**	0.880	83	0.890	166
MIWrapper (Bagging&PART)	**0.900**	**41**	0.850	83	0.870	166
MIWrapper (PART)	**0.930**	**41**	0.810	83	0.820	166
MIWrapper (SMO)	**0.780**	**66**	0.710	83	0.740	166
MIWrapper (Naive Bayes)	**0.790**	**54**	0.730	83	0.760	166
MISimple (AdaBoost&PART)	**0.890**	**46**	0.830	83	0.820	166
MISimple (PART)	**0.860**	**57**	0.760	83	0.800	166

novel examples by retrieving the nearest stored training example, using all the available features in its distance computations (such as, CitationKNN and MIOptimaBall), which improve considerably. Towards the other extreme lie algorithms that explicitly try to focus on relevant features and ignore irrelevant ones. Methods based on Diverse Density are examples of this approach (for example, the MDD, MIDD, MIEMDD) whose improvement is less noticeable.

Finally, a statistical study is carried out to determine whether algorithms using our HyDR-MI hybrid to reduce the dimensionality presents better results than the ReliefF-MI method alone and the consideration of all the features. The idea is to verify, in general, if there are significant differences between the accuracy values obtained by different algorithms using the feature set provided by the HyDR-MI algorithm, the feature set provided by ReliefF-MI, or the use of all features directly. The Wilcoxon rank-sum test is used to look for differences between the accuracy values obtained by the two methods. This test is a non-parametric one, recommended in the study of Garcia et al. (15): it allows us to address the question of whether there are significant differences between the accuracy values obtained by algorithms when using different feature sets with the two methods. To do this, the null hypothesis of this test maintains that there are not significant differences between the accuracy values obtained by the algorithms when they use different feature sets, while the alternative hypothesis asserts that there are. So, this test evaluates the differences in performance in the two methods, evaluating the results obtained by the algorithms when they use different feature sets, from the most reduced to the use of all the features. Table 12 shows the mean ranks and the sum of ranks for each comparison and dataset. For each dataset, there is a comparison between HyDR-MI and the other methods. The scores are ranked from lowest to highest. As a result, algorithms using all features and the subset provided by ReliefF-MI are seen to place at a lower mean rank than algorithms using the HyDR-MI method. This information can be used to ascertain a priori that HyDR-MI improves the results.

Table 13 contains the results of the Wilcoxon test and its corresponding *z*-score. Moreover, the significance value of the test is shown, which gives a two-tailed probability to determine acceptance or rejection of the null hypothesis. According to this value, the results are highly significant (p-value < 0.01 or p-value < 0.05). Therefore, at a 99%/95% confidence level (de-

Table 12: Sum of ranks and mean rank of the two methods

Dataset		Method	Mean Rank	Sum of Ranks
Fox	Comparison1	HyDR-MI	**22.56**	**383.50**
		ReliefF-MI	12.44	211.50
	Comparison2	HyDR-MI	**22.50**	**382.50**
		Full Features	12.50	212.50
Elephant	Comparison1	HyDR-MI	**22.47**	**382.00**
		ReliefF-MI	12.53	213.00
	Comparison2	HyDR-MI	**21.74**	**369.50**
		Full Features	13.26	225.50
Tiger	Comparison1	HyDR-MI	**23.44**	**398.50**
		ReliefF-MI	11.56	196.50
	Comparison2	HyDR-MI	**23.88**	**406.00**
		Full Features	11.12	189.00
Musk1	Comparison1	HyDR-MI	**22.18**	**377.00**
		ReliefF-MI	12.85	218.00
	Comparison2	HyDR-MI	**21.03**	**357.50**
		Full Features	13.97	237.50
Musk2	Comparison1	HyDR-MI	**22.29**	**379.00**
		ReliefF-MI	12.71	216.00
	Comparison2	HyDR-MI	**21.71**	**369.00**
		Full Features	13.29	226.00

Table 13: Wilcoxon rank-sum test results

Dataset		Wilcoxon W	Z-score	Asymp sig (2-tailed) (p-value)
Fox	HyDR-MI vs ReliefF-MI	211.50	-2.966	0.003
	HyDR-MI vs Full Features	212.50	-2.929	0.003
Elephant	HyDR-MI vs ReliefF-MI	213.00	-2.913	0.004
	HyDR-MI vs Full Features	225.50	-2.483	0.013
Tiger	HyDR-MI vs ReliefF-MI	196.00	-3.485	0.000
	HyDR-MI vs Full Features	189.00	-3.739	0.000
Musk1	HyDR-MI vs ReliefF-MI	218.00	-2.742	0.006
	HyDR-MI vs Full Features	237.50	-2.068	0.039
Musk2	HyDR-MI vs ReliefF-MI	216.00	-2.814	0.005
	HyDR-MI vs Full Features	226.00	-2.469	0.014

pending on the comparison), the null hypothesis is rejected for all datasets and there are significant differences between the results obtained by algorithms when they use different methods of dimensionality reduction. Consequently, HyDR-MI has significantly higher accuracy values than using all features or ReliefF-MI. This conclusion is reached by noting that for HyDR-MI scores, the mean rank is higher in the algorithms using this method to pre-process their features (for example, for the fox dataset at a value of 22.50) than when using all features (at a value of 12.50).

5. Conclusion

Feature selection is a common approach for dimensionality reduction in supervised learning. Feature selection techniques eliminate those features that are not relevant for solving the problem, thus allowing classification techniques to improve their accuracy and speed. Although many feature selection approaches have been proposed for traditional supervised learning setting, relatively little has been done in this area for MIL settings. In this study, we proposed a framework that combines the advantages of the filter and wrapper algorithms.

The filter model is based on the ReliefF-MI method that adapts the ideas of the ReliefF algorithm to MIL settings, and in the wrapper we employ GA-based search for the best feature subset from the reduced set of features output by the filter component. To determine a suitable configuration of the hybrid we developed and experimentally evaluated several alternatives.

The experimental results compared the performance of the proposed hybrid algorithm in terms of its accuracies. The achieved results are very promising as HyDR-MI helped to a considerable number of algorithms to improve their results, including methods based on support vector machines, rules, logistic regression, distance, probabilistic, and so on. Specifically, statistical tests show that the results are really improved considering the use of all features and the filter component to reduce features.

Acknowledgment

The authors gratefully acknowledge the subsidy provided by the Spanish Department of Research under TIN2008- 06681-C06-03, P08-TIC-3720 Projects and FEDER funds

References

[1] S. Ali, M. Shah, Human action recognition in videos using kinematic features and multiple instance learning, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2) (2010) 288–303.

[2] S. Andrews, I. Tsochantaridis, T. Hofmann, Support vector machines for multiple-instance learning, in: NIPS'02: Proceedings of Neural Information Processing System, 2002, pp. 561–568.

[3] P. Auer, R. Ortner, A boosting approach to multiple instance learning, in: ECML'04: Proceedings of the 5th European Conference on Machine Learning. Lecture Notes in Computer Science, vol. 3201, 2004, pp. 63–74.

[4] H. Blockeel, D. Page, A. Srinivasan, Multi-instance tree learning, in: ICML '05: Proceedings of the 22nd international conference on Machine learning, ACM, New York, 2005, pp. 57–64.

[5] Y.-M. Chai, Z.-W. Yang, A multi-instance learning algorithm based on normalized radial basis function network, in: ISSN'07: Proceedings of the 4th International Symposium on Neural Networks. Lecture Notes in Computer Science, vol. 4491, Nanjing, China, 2007, pp. 1162–1172.

[6] X. Chen, C. Zhang, S. Chen, S. Rubin, A human-centered multiple instance learning framework for semantic video retrieval, IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews 39 (2) (2009) 228–233.

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

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

[9] Y.-Z. Chevaleyre, J.-D. Zucker, Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. Application to the mutagenesis problem, in: AI'01: Proceedings of the 14th of the Canadian Society for Computational Studies of Intelligence, Lecture Notes in Computer Science, vol. 2056, Ottawa, Canada, 2001, pp. 204–214.

[10] H. Cohen, Image restoration via n-nearest neighbour classification, in: ICIP'96: Proceedings of the International Conference on Image Processing, 1996, pp. 1005–1007.

[11] K. De Jong, Learning with Genetic Algorithms: An overview. Volumen 3, Kluwer, 1998.

[12] T. G. Dietterich, R. H. Lathrop, T. Lozano-Perez, Solving the multiple instance problem with axis-parallel rectangles, Artifical Intelligence 89 (1-2) (1997) 31–71.

[13] O. J. Dunn, Multiple comparisons among means, Journal of the American Statistical Association 56 (293) (1961) 52–64.

[14] G. Edgar, Measure, Topology, and Fractal Geometry, Third Edition, Springer-Verlag, Berlin, 1995.

[15] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Information Sciences 180 (2010) 2044–2064.

[16] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.

[17] T. Gärtner, P. A. Flach, A. Kowalczyk, A. J. Smola, Multi-instance kernels, in: ICML'02: Proceedings of the 19th International Conference on Machine Learning, Morgan Kaufmann, 2002, pp. 179–186.

[18] Z. Gu, T. Mei, J. Tang, X. Wu, X. Hua, MILC2: A multi-layer multi-instance learning approach to video concept detection, in: MMM'08: Proceedings of the 14th International Conference of Multimedia Modeling, Kyoto, Japan, 2008, pp. 24–34.

[19] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.

[20] G. Herman, G. Ye, J. Xu, B. Zhang, Region-based image categorization with reduced feature set, in: Proceedings of the 10th IEEE Workshop on Multimedia Signal Processing, 2008, pp. 586–591.

[21] Y. Huang, P. J. McCullagh, N. D. Black, An optimization of ReliefF for classification in large datasets, Data and Knowledge Engineering 68 (2009) 1348–1356.

[22] S. Keerthi, S. Shevade, C. Bhattacharyya, K. Murthy, Improvements to Platt's SMO algorithm for SVM classifier design, Neural Computation 13 (3) (2001) 637–649.

[23] I. Kononenko, Estimating attributes: Analysis and extension of relief, in: ECML'94: Proceedings of the 7th European Conference in Machine Learning, Springer-Verlag, Berlin, 1994, pp. 171–182.

[24] L. Kuncheva, L. Jain, Nearest neighbor classifier: Simultaneous editing and feature selection, Pattern Recognition Letters 20 (1999) 1149–1156.

[25] S. Maldonado, R. Weber, J. Basak, Simultaneous feature selection and classification using kernel-penalized support vector machines, Information Sciences 181 (1) (2011) 115–128.

[26] O. L. Mangasarian, E. W. Wild, Multiple instance classification via successive linear programming, Journal of Optimization Theory and Applications 137 (3) (2008) 555–568.

[27] O. Maron, T. Lozano-Pérez, A framework for multiple-instance learning, in: NIPS'97: Proceedings of Neural Information Processing System 10, MIT Press, Cambridge, MA, USA, 1997, pp. 570–576.

[28] I.-S. Oh, J.-S. Lee, Hybrid genetic algorithms for feature selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (11) (2004) 1424–1437.

[29] J. Pang, Q. Huang, S. Jiang, Multiple instance boost using graph embedding based decision stump for pedestrian detection, in: ECCV'08: Proceedings of the 10th European Conference on Computer Vision, No. 4 in LNCS 5305, Springer-Verlag, Berlin, 2008, pp. 541–552.

[30] H. Pao, S. Chuang, Y. Xu, H. Fu, An EM based multiple instance learning method for image classification, Expert Systems with Applications 35 (3) (2008) 1468–1472.

[31] X. Qi, Y. Han, Incorporating multiple svms for automatic image annotation, Pattern Recognition 40 (2) (2007) 728–741.

[32] S. Ray, M. Craven, Supervised versus multiple instance learning: An empirical comparison, in: ICML'05: Proceedings of the 22nd international conference on Machine learning, ACM, New York, 2005, pp. 697–704.

[33] V. C. Raykar, B. Krishnapuram, J. Bi, M. Dundar, R. B. Rao, Bayesian multiple instance learning: automatic feature selection and inductive transfer, in: ICML '08: Proceedings of the 25th international conference on Machine learning, ACM, New York, 2008, pp. 808–815.

[34] M. Raymer, W. Punch, E. Goodman, L. Kuhn, A. Jain, Dimensionality reduction using genetic algorithms, IEEE Transactions on Evolutionary Computation 4 (2) (2000) 164–171.

[35] O. Ritthoff, R. Klinkenberg, S. Fischer, I. Mierswa, A hybrid approach to feature selection and generation using an

evolutionary algorithm, UK, 2002, pp. 147–154.

[36] G. Ruffo, Learning single and multiple instance decision tree for computer security applications, Master's thesis, Department of Computer Science. University of Turin, Turin, Italy (2000).

[37] S. Scott, J. Zhang, J. Brown, On generalized multiple-instance learning, International Journal of Computational Intelligence and Applications 5 (2005) 21–35.

[38] W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, Pattern Recognition Letters 10 (1989) 335–347.

[39] Q. Tao, S. Scott, N. V. Vinodchandran, T. T. Osugi, SVM-based generalized multiple-instance learning via approximate box counting, in: ICML'04: Proceedings of the 21st international conference on Machine learning, ACM, New York, 2004, pp. 799–806.

[40] A. Tsymbal, M. Pechenizkiy, P. Cunningham, Diversity in search strategies for ensemble feature selection., Information Fusion 6 (1) (2005) 83–98.

[41] J. Wang, J.-D. Zucker, Solving the multiple-instance problem: A lazy learning approach, in: ICML'00: Proceedings of the 17th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, 2000, pp. 1119–1126.

[42] N. Weidmann, E. Frank, B. Pfahringer, A two-level learning method for generalized multi-instance problems, in: ECML'03: Proceedings of the 14th European Conference on Machine Learning, 2003, pp. 468–479.

[43] I. H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques. Second Edition, Morgan Kaufmann, San Francisco, 2005.

[44] X. Xu, E. Frank, Logistic regression and boosting for labeled bags of instances, in: PAKDD'04: Proceedings of the 8th Conference of Pacific-Asia. Lecture Notes in Computer Science, vol. 3056, 2004, pp. 272–281.

[45] C. Yang, M. Dong, F. Fotouhi, Region based image annotation through multiple-instance learning, in: Multimedia'05: Proceedings of the 13th Annual ACM International Conference on Multimedia, New York, 2005, pp. 435–438.

[46] C. Yang, T. Lozano-Perez, Image database retrieval with multiple-instance learning techniques, in: ICDE '00: Proceedings of the 16th International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, 2000, pp. 233–243.

[47] X. Yuan, X.-S. Hua, M. Wang, G.-J. Qi, X.-Q. Wu, A novel multiple instance learning approach for image retrieval based on adaboost feature selection, in: ICME'07: Proceedings of the IEEE International Conference on Multimedia and Expo, IEEE, Beijing, China, 2007, pp. 1491–1494.

[48] A. Zafra, E. Gibaja, S. Ventura, Multi-instance learning with multi-objective genetic programming for web mining, Applied Soft Computing 11 (2011) 93–102.

[49] A. Zafra, M. Pechenizkiy, S. Ventura, Reducing dimensionality in multiple instance learning with a filter method, in: HAIS'10: Proceedings of the 5th International Conference on Hybrid Artificial Intelligence Systems, vol. 6077 of Lecture Notes in Computer Science, San Sebastián, Spain, 2010, pp. 35–44.

[50] A. Zafra, S. Ventura, Predicting student grades in learning management systems with multiple instance genetic programming, in: EDM'09: Proceedings of the 2nd Conference on Educational Data Mining, Cordoba, Spain, 2009, pp. 309–319.

[51] A. Zafra, S. Ventura, G3P-MI: a genetic programming algorithm for multiple instance learning, Information Sciences 180 (23) (2010) 4496–4513.

[52] A. Zafra, S. Ventura, C. Romero, E. Herrera-Viedma, Multi-instance genetic programming for web index recommendation, Expert System with Applications 36 (2009) 11470–11479.

[53] D. Zhang, F. Wang, L. Si, T. Li, M3IC: Maximum margin multiple instance clustering, in: International Joint Conference on Artificial Intelligence, 2009, pp. 1339–1344.

[54] M.-L. Zhang, Z.-H. Zhou, Improve multi-instance neural networks through feature selection, Neural Processing Letter 19 (1) (2004) 1–10.

[55] M.-L. Zhang, Z.-H. Zhou, Ensembles of multi-instance Neural Networks, in: IIP'04: International Conference on Intelligent Information Processing II. IFIP International Federation for Information Processing, vol. 163, Beijing, China, 2005, pp. 471–474.

[56] M.-L. Zhang, Z.-H. Zhou, Adapting RBF Neural Networks to multi-instance learning, Neural Processing Letters 23 (1) (2006) 1–26.

[57] M.-L. Zhang, Z.-H. Zhou, Multi-instance clustering with applications to multi-instance prediction, Applied Intelligence 31 (2009) 47–68.

[58] Q. Zhang, S. Goldman, EM-DD: An improved multiple-instance learning technique, in: NIPS'01: Proceedings of Neural Information Processing System 14, Vancouver, Canada, 2001, pp. 1073–1080.

[59] Z.-H. Zhou, Multi-instance learning from supervised view, Journal Computer Science and Technology 21 (5) (2006) 800–809.

[60] Z.-H. Zhou, K. Jiang, M. Li, Multi-instance learning based web mining, Applied Intelligence 22 (2) (2005) 135–147.

[61] Z.-H. Zhou, J.-M. Xu, On the relation between multi-instance learning and semi-supervised learning, in: ICML'07: Proceedings of the 24th international conference on Machine learning, Corvallis, OR, USA, 2007, pp. 1167–1174.
[62] Z.-H. Zhou, M.-L. Zhang, Ensembles of multi-instance learners, in: ECML'03: Proceedings of the 14th European Conference on Machine Learning, vol. 2837 of Lecture Note in Artifical Intelligence, 2003, pp. 492–502.
[63] Z.-H. Zhou, M.-L. Zhang, Solving multi-instance problems with classifier ensemble based on constructive clustering, Knowledge and Information Systems 11 (2) (2007) 155–170.
[64] Z. Zhu, Y.-S. Ong, M. Dash, Wrapperfilter feature selection algorithm using a memetic framework, IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 37 (1) (2007) 70–76.