Artificial Intelligence 201 (2013) 81–105

Contents lists available at SciVerse ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Multiple instance classification: Review, taxonomy and comparative study

Jaume Amores

Computer Vision Center, Computer Science Department, UAB, Spain

ARTICLE INFO

Article history:
Received 14 July 2011
Received in revised form 24 May 2013
Accepted 6 June 2013
Available online 19 June 2013

Keywords:
Multi-instance learning
Codebook
Bag-of-Words

ABSTRACT

Multiple Instance Learning (MIL) has become an important topic in the pattern recognition community, and many solutions to this problem have been proposed until now. Despite this fact, there is a lack of comparative studies that shed light into the characteristics and behavior of the different methods. In this work we provide such an analysis focused on the classification task (i.e., leaving out other learning tasks such as regression). In order to perform our study, we implemented fourteen methods grouped into three different families. We analyze the performance of the approaches across a variety of well-known databases, and we also study their behavior in synthetic scenarios in order to highlight their characteristics. As a result of this analysis, we conclude that methods that extract global bag-level information show a clearly superior performance in general. In this sense, the analysis permits us to understand why some types of methods are more successful than others, and it permits us to establish guidelines in the design of new MIL methods.

1. Introduction

In the standard supervised learning task, we learn a classifier based on a training set of feature vectors, where each feature vector has an associated class label. In the Multiple Instance Learning (MIL) task we learn a classifier based on a training set of *bags*, where each bag contains multiple feature vectors (called instances in the MIL terminology). In this setting, each bag has an associated label, but we do not know the labels of the individual instances that conform the bag. Furthermore, not all the instances are necessarily relevant, i.e., there might be instances inside one bag that do not convey any information about its class, or that are more related to other classes of bags, providing confusing information.

In many fields, we find problems that are most naturally formulated using the multiple instance learning setting. This is the case of drug discovery (pharmacy), classification of text documents (information retrieval), classification of images (computer vision), speaker identification (signal processing) and bankruptcy prediction (economy), to mention a few fields that make use of this framework (see Section 2 for a more detailed discussion about real examples). This makes the MIL problem an important topic in the machine learning community, where many methods have been published in the last years. Despite this fact, there is a lack of surveys or analytical studies that compare the performance of the different families of MIL algorithms.

In this work, we focus on Multiple Instance Classification (MIC), leaving out other learning tasks such as regression. We present an extensive review of the methods of the literature accompanied with a thorough empirical comparison. In our analysis, we grouped the methods into a small set of compact paradigms according to how they manage the information from the Multi-Instance (MI) data. As we will see, our characterization is complete in the sense that any MIC method must necessarily fall into one of the families of the proposed taxonomy. Furthermore, the methods falling into each paradigm

E-mail address: jaume@cvc.uab.es.

0004-3702/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.artint.2013.06.003

tend to present a similar behavior, and this makes it easy to analyze and compare the paradigms in the experimental evaluation. As part of the proposed taxonomy we characterize for the first time the Vocabulary-based paradigm. The main difference between this and other paradigms is that in the Vocabulary-based one the instances are classified or discriminated into several classes, while in the other paradigms there is no such discrimination. Many authors [1–9] have proposed algorithms that fall into the Vocabulary-based paradigm, but the relationship between all these approaches has not been established until now. In this work, we show that all of them fall under the Vocabulary-based family and we provide a clear characterization of this family.

We are only aware of the recent review by Foulds and Frank [10], and the comparative study performed in the master's thesis of Lin Dong [11]. Unfortunately, these publications do not include the family of Vocabulary-based techniques as such, which is an important paradigm as we show in this paper. In [11], Lin Dong shows a quantitative analysis of many methods but does not obtain conclusive results and leaves out many important algorithms from the Vocabulary-based paradigm. Recently, Foulds and Frank [10] categorized the MIC methods according to the assumption followed by each one. As we show in this work, many categories of methods proposed in [10] fall into the Vocabulary-based one characterized in our work. In our work, we present a complementary analysis in the sense that Foulds and Frank classify the methods according to the assumption followed by each one, while we perform this classification according to the type of information extracted by each method (instance-level or bag-level information) and how it is represented (implicitly or explicitly). In this sense, our analysis is not in conflict with [10]. Furthermore, we provide an empirical evaluation of the proposed paradigms and analyze their behavior, which is not done in [10].

In summary, this work contributes a novel analysis and taxonomy of the MIC methods and an exhaustive comparative analysis. In total, we analyze fourteen MIC algorithms implemented by us, and we use eight databases from four different fields of knowledge, plus a synthetic database where we studied the behavior of the methods under controlled conditions. A preliminary version of this work appeared in [12].

The rest of the paper is organized as follows. In Section 2 we motivate in detail the need of using Multiple Instance Classification through real examples. In Section 3 we describe the MIC problem and the taxonomy proposed. In Sections 4, 5 and 6 we describe the main paradigms of the taxonomy: the instance-space paradigm, the bag-space paradigm and the embedded-space paradigm. The latter paradigm contains the Vocabulary-based family of methods, which is described in detail in Section 7. Section 9 provides a comparative analysis of the different paradigms, and we present conclusions in Section 10.

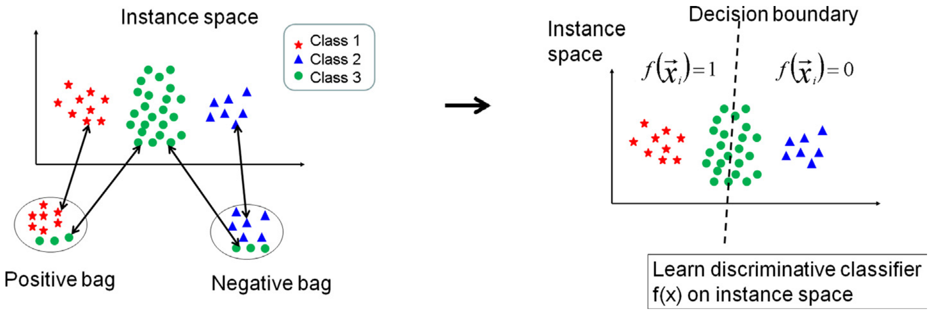
2. Examples of multiple-instance classification problems

We describe here two real problems where the MI representation becomes necessary, i.e., where the objects to be classified are described by bags (containing multiple feature vectors) as opposed to the traditional learning problem where the objects to be classified are represented by means of a single feature vector.

The first one is the drug activity prediction problem [13]. In this problem, the objects to be classified are chemical molecules. Given a molecule, the system must decide if it is a good drug or it is not. A good drug is characterized by the fact that it is able to bind strongly to a target "binding site", which is some sort of cavity existing in a much larger molecule [13]. The difficulty comes from the fact that one molecule can adopt multiple three-dimensional shapes (called conformations), and only one or a few of them bind well with the target binding site. In this type of problem, the complete molecule is described by a bag $X = \{\vec{x}_1, \ldots, \vec{x}_N\}$, which is a set that gathers the description of the N possible conformations, where $\vec{x}_i$, $i = 1, \ldots, N$, is a feature vector describing the i-th conformation, and the number of conformations N can vary in different molecules.

Another example of a real problem where MIC becomes important is the one of image classification. Here, given an image we must decide if it belongs to a target class, based on its visual content. For example, the target class might be "beach", and in this case the positive images are those displaying a beach, while the negative images will be those displaying any other type of visual content. Fig. 6 shows an example of this image classification task, where we explain later the meaning of the red circles. In this figure, the images in the top row are positive, while the images in the bottom row are negative (one of them displays the sea, but without any beach in it, while the other image displays a desert). Although we only show two negative images in this figure, there are many other negative images which contain any other type of content such as countryside, cities, cars, offices, etc. If we look at the positive images in the top row of Fig. 6, we can see that there are regions of the image that are related with the target class (the regions that belong to the sand and sea), whereas there are regions that are not specifically related with it (e.g., the sky, mountain, trees, etc.). In order to obtain a beach image we need both sea and sand, while the rest of regions are not necessary. In order to classify the images, the usual procedure is to first extract a collection of regions in the image, and for each region we obtain a visual descriptor. This visual descriptor is a feature vector that describes the region. As a result, the image is described as a bag $X = \{\vec{x}_1, \ldots, \vec{x}_N\}$, where N is the number of regions extracted and $\vec{x}_i$ is the feature vector (called instance) describing the i-th region in the image. In Fig. 6 we use red circles for symbolizing the extraction of visual descriptors in different regions. The number of regions extracted depends on the specific algorithm for identifying interesting regions, and might vary from image to image.

These are just two examples of real problems where using a bag representation, and hence setting the problem as MIC, is necessary. In addition to these two problems, there are many other problem domains that require this type of formu-



(a) (b)

Fig. 1. Illustration of the IS paradigm, see text.

lation, as mentioned in the introduction, including classification tasks in information retrieval, audio processing, economic predictions, etc. In the rest of this paper we study the different approaches for solving MIC problems.

3. Basic concepts and overview of paradigms

A bag is a set $X=\{\vec{x}_1,\dots,\vec{x}_N\}$, where the elements $\vec{x}_i$ are feature vectors called *instances* in the MIC terminology, and the cardinality N can vary across the bags. All the instances $\vec{x}_i$ live in a d-dimensional feature space, $\vec{x}_i\in\mathbb{R}^d$, called *instance space*.

The objective of the MIC problem is to learn a model, at training time, that can be used to predict the class labels of unseen bags. In this work, we only consider the binary classification problem, where a bag X can be either positive or negative. Our objective is to estimate a classification function $F(X)\in[0,1]$ that provides the likelihood that X is positive. In order to learn such a function, we are given a training set with M bags and their corresponding labels, $\mathcal{T}=\{(X_1,y_1),\dots,(X_M,y_M)\}$, where $y_i\in\{0,1\}$ is the label of X_i ($y_i=0$ if X_i is negative, and $y_i=1$ if it is positive).

In addition to the bag-level classification function $F(X)$, many methods try to learn an instance-level classification function $f(\vec{x}_i)$ that operates directly on the instances $\vec{x}_i$. Throughout this work we will use uppercase to refer to bags X and to the bag-level classifier F, and we will use lowercase to refer to instances $\vec{x}$ and to the instance-level classifier f.

3.1. Overview of proposed taxonomy

In this work, we categorize the MIC methods according to how the information existent in the MI data is exploited (see Fig. 4). In the Instance-Space (IS) paradigm, the discriminative information is considered to lie at the instance-level. Therefore, the discriminative learning process occurs at this level: a discriminative *instance-level* classifier $f(\vec{x})$ is trained to separate the instances in positive bags from those in negative ones (see Fig. 1). Based on it, given a new bag X the bag-level classifier $F(X)$ is obtained by simply aggregating instance-level scores $f(\vec{x})$, $\vec{x}\in X$. We say that this type of paradigm is based on *local*, instance-level information, in the sense that the learning process considers the characteristics of individual instances, without looking at more global characteristics of the whole bag.

In the Bag-Space (BS) paradigm, the discriminative information is considered to lie at the bag-level. In this paradigm each bag X is treated as a whole entity, and the learning process discriminates between entire bags. As a result, it obtains a discriminative bag-level classifier $F(X)$ which makes use of the information from the whole bag X in order to take a discriminative decision about the class of X. We say that this type of paradigm is based on *global*, bag-level information, because the discriminative decision is taken by looking at the whole bag, instead of aggregating local instance-level decisions.

Given the fact that the bag space is a non-vector space, the BS methods make use of non-vectorial learning techniques. As far as we know, all the existent non-vectorial techniques work through the definition of a distance function $D(X,Y)$ that provides a way of comparing any two non-vectorial entities X and Y (where these entities are bags in our problem). Once this distance function has been defined, it can be used into any standard distance-based classifier such as K-Nearest Neighbor (K-NN), or similarly into any kernel-based classifier such as SVM.[1] Fig. 2 illustrates the idea under this paradigm. Although we use the term “distance” in Fig. 2, the BS paradigm also includes methods that use other types of pairwise comparisons between bags, such as kernel-based comparisons $K(X,Y)$ in SVM-based methods. Regarding the bag-level classifier, we use the notation $F(X;\Theta)$ in Fig. 2(b), in order to express the fact that the classifier makes use of the learned parameters Θ (see Fig. 2(a)). Along the paper, however, we use the notation $F(X)$ and drop the argument Θ for simplicity.

[1] As we will see, any distance function $D(X,Y)$ can be transformed into a kernel function $K(X,Y)$. Similarly, any kernel function can be transformed into a distance function.

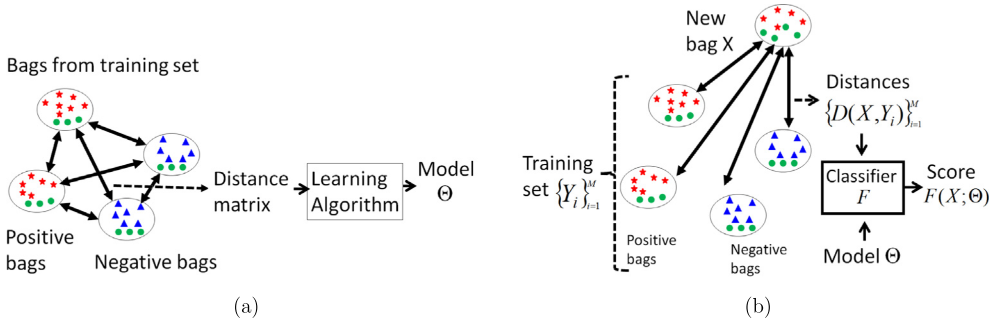


Fig. 2. Illustration of the BS paradigm: training (a) and test (b). See text for an explanation.

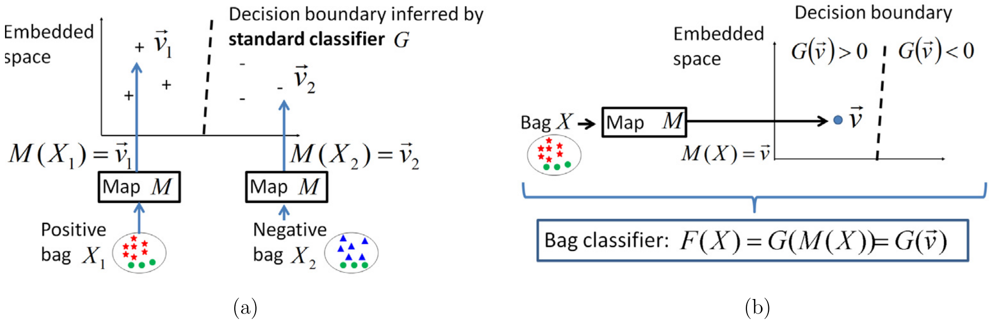


Fig. 3. Illustration of the ES paradigm: training (a) and test (b). See text.

In the Embedded-Space (ES) paradigm, each bag X is mapped to a single feature vector which summarizes the relevant information about the whole bag X. As a result, the original bag space is mapped to a *vectorial* embedded space, where the discriminative classifier is learned. This effectively transforms the original MIC problem into a standard supervised learning problem, where each feature vector has an associated label and any standard classifier such as AdaBoost, Neural Networks or SVM can be applied. Fig. 3 illustrates the idea under this paradigm.

Note that the ES paradigm is also based on global, bag-level information, in the sense that the bag X is represented by a feature vector $\vec{v}$ that summarizes the relevant information about the whole bag. Given this feature vector, the bag-level classifier $F(X)$ can be expressed as $F(X) = G(\vec{v})$, where G is a discriminant classifier that makes it decision based on the vector $\vec{v}$ summarizing the whole bag.

In this sense, both the ES and the BS paradigms exploit global, bag-level information. However, the difference between both paradigms lies in the way this bag-level information is extracted. In the BS paradigm, this is done *implicitly* through the definition of a distance or kernel function.[2] In contrast, in the ES paradigm, the extraction of information from the whole bag is performed *explicitly* through the definition of a mapping function that defines how the relevant information is represented into a single vector $\vec{v}$.

Therefore, we categorize the methods based on whether they focus on instance-level information (IS paradigm) or global, bag-level information, and in the last case whether they extract the relevant information implicitly (BS paradigm) or explicitly (ES paradigm). In addition to this, there is also a characteristic computational cost for each paradigm.

3.2. Completeness of the proposed categorization

The presented categorization is complete in the sense that, given any MIC method from the literature it must necessarily fall into one of the three families: IS, BS or ES. If the MIC method obtains the bag-level classification $F(X) \in [0,1]$ as an aggregation of instance-level classifications $f(\vec{x}) \in [0,1]$ for all $\vec{x} \in X$, then it falls into the IS paradigm. Otherwise, the method falls either in the BS or the ES paradigm. In the latter case, if the bag X is mapped into a feature vector $\vec{v}$ and then classified by any standard classifier, then it belongs to the ES paradigm. Otherwise, if no such mapping is applied and the bag is classified as a whole, we have a BS method.

In any categorization we will always find methods that fall close to the boundaries of two categories. For example, in our taxonomy this happens with the BARTMIP method (see Section 7.6). This is an ES method where each bag X is explicitly mapped into a vector $\vec{v}$. However, in order to perform this mapping, the bag X is compared with other bags Y from the

[2] Indeed, a kernel function $K(X,Y)$ defines an implicit mapping $\phi(X) \mapsto \vec{v}$. The function ϕ maps the original bag space (where the bag X lives) into a new vector space, where the vector $\vec{v}$ lives [14].

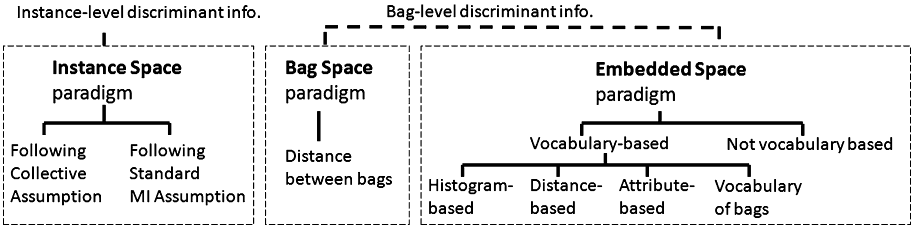


Fig. 4. Proposed taxonomy of MIC methods.

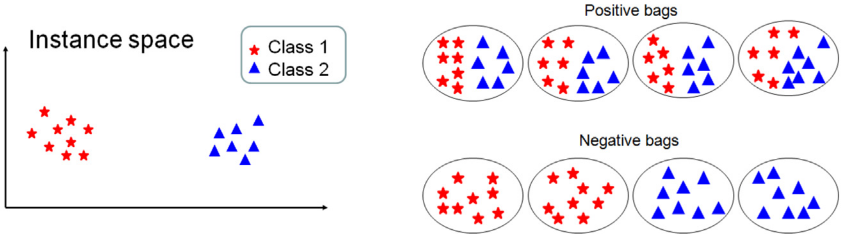


Fig. 5. Illustrative toy example of an MIC problem where *global* bag-level information becomes necessary. See text.

training set through the definition of a bag-level distance function $D(X, Y)$. In this sense, this method lies close to the boundary between the ES and BS categories.

3.3. Illustrative examples

In the rest of the work we will use two illustrative synthetic examples. The first one is shown in Fig. 1, and illustrates the case where instance-level information is enough for solving the MIC problem. This happens when there are certain classes of instances that appear only in positive bags, so that learning an (instance-level) model of these classes is enough. In particular, in the example illustrated in Fig. 1 the class of instances 1 only appears in positive bags. Therefore, it is enough to learn an instance-level classifier $f(\vec{x}) \in [0, 1]$ that provides the confidence that instance $\vec{x}$ belongs to class 1. Once $f(\vec{x})$ has been learned, the bag-level classifier $F(X)$ can be simply obtained by taking the maximum over the instance-level scores: $F(X) = \max_{\vec{x} \in X} f(\vec{x})$. This way, the bag X is classified as positive if any of its instances $\vec{x} \in X$ belongs to class 1, and classified as negative otherwise.

Note that by using such an approach, the learning is performed only at the *instance-level*, i.e., for obtaining a model of the instances of class 1 which is used by the instance-level classifier $f(\vec{x})$. At the bag-level, however, there is no learning, as the classifier $F(X)$ is obtained as an aggregation of instance-level scores. This type of approach forms part of the IS paradigm, which is characterized in Section 4, and it works for MIC problems such as the one in Fig. 1, where there is at least one class of instances that appears only in positive bags (or the other way around, there is at least one class of instances that appears only in negative bags).

Fig. 5 shows another synthetic MIC problem where this does not happen. Here, there are two classes of instances, and both of them appear in positive and negative bags. Hence, there is no single class of instances that appears only in positive or only in negative bags. In this type of problem, the learning cannot be performed *only* at the instance-level. For example, if we learn an instance-level model of class 1 in order to obtain $f(\vec{x})$, then we cannot infer the classification $F(X)$ based only on the individual scores $f(\vec{x})$, as we find that both positive and negative bags contain instances of class 1. The same happens if we learn an instance-level model of class 2.

If we look at the composition of the bags in Fig. 5, we find that positive bags are characterized by containing instances of both class 1 *and* class 2. In contrast, negative bags are characterized by containing *either* instances of class 1 *or* instances of class 2, but not both of them at the same time. Therefore, it is not enough to learn an (instance-level) model of the classes of instances, but we must learn a bag-level model about the composition of the whole bag. As we will see, this bag-level information can be learned if we use a BS or ES method. Indeed, we will see that both BS and ES are successful in both the problem shown in Fig. 1 and the one in Fig. 5, while IS methods only succeed in the first type of problem.

Fig. 6 illustrates a real MIC problem similar to the one in Fig. 5. The problem concerns the classification of images into beach (top row images) and non-beach (bottom row images). Here, each image is described as a bag of instances, where the i-th instance $\vec{x}_i$ describes the i-th local region of the image. The idea is symbolized in the images by using red circles, each one corresponding to one local region. In this type of MIC problem, each bag contains several classes of instances, depending on the regions that conform the image. In order for the image to belong to the class *beach*, instances of class *sand* and class *sea* must co-occur. However, if only one of these classes occurs in the image then the class is *non-beach*. This type of MI data happens rather frequently in MIC problems, not only in image classification tasks. In addition to this, we will also discuss some other examples where a global bag-level approach to classification is fundamental.

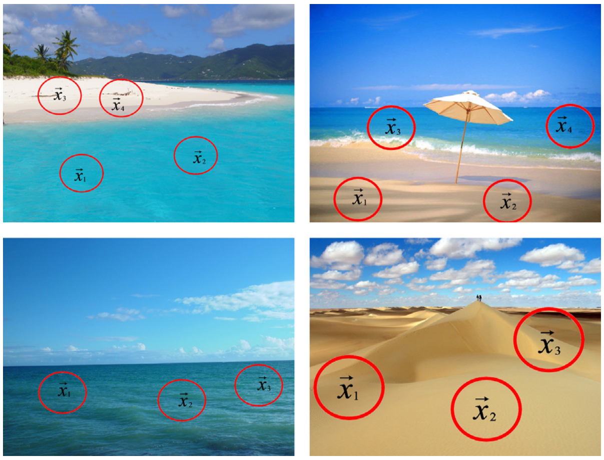


Fig. 6. Classification of images into beach (top row) and non-beach (bottom row). See text.

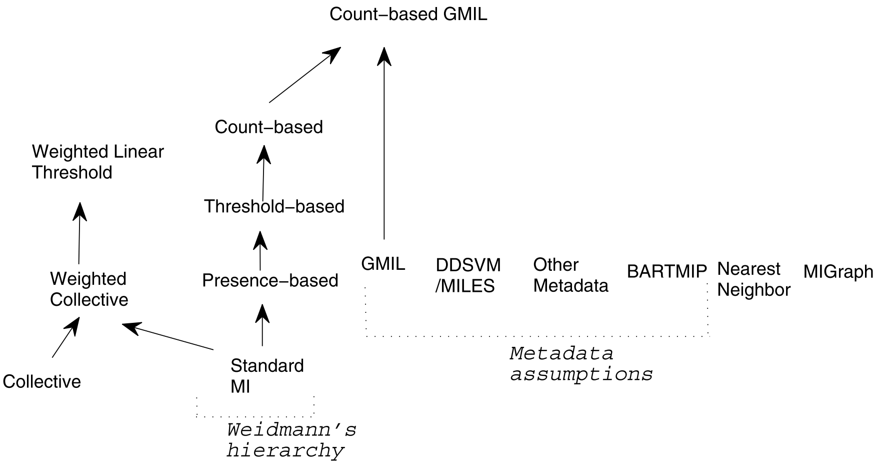


Fig. 7. Foulds and Frank MIC taxonomy according to the followed assumption [10].

3.4. Related work

In Fig. 7, we show the hierarchy of categories proposed in Foulds and Frank [10]. Comparing Figs. 4 and 7, we see that [10] divides the methods into more paradigms, some of them disconnected from the rest. As we will see, the majority of the paradigms which are isolated in Fig. 7 are indeed part of the Vocabulary-based family, which is characterized for the first time in our review and extensively analyzed.

Note that Foulds and Frank obtain their taxonomy using a different underlying criteria. In their case, they pay attention to the assumption which each method uses about the relationship between bag labels and instances. Note that some assumptions were stated explicitly by the author of each method, and some others were not, so that they can only be guessed from the algorithm. In our case, we use as criteria at what level the *discriminant* information is extracted and how it is represented. The two criteria do not conflict and they both help obtain a deeper understanding of the different MIC solutions. Note also that the objectives of both taxonomies are not necessarily the same. In our case the proposed taxonomy has two objectives: first, to provide a clear picture about the existing approaches, and second, to allow an experimental analysis of the methods, in such a way that methods can be compared according to the paradigm they belong to. This last objective is not necessarily pursued by Foulds and Frank, as they do not provide a comparative analysis.

We proceed now to describe in detail the IS paradigm (Section 4), the BS paradigm (Section 5), and the ES one (Section 6).

4. Instance-space paradigm

As explained in Section 3, in the IS paradigm the idea is to infer an instance-based classifier $f(\vec{x}) \in [0, 1]$ from the training data. Based on this classifier, the bag-level classification $F(X) \in [0, 1]$ is constructed as an aggregation of instance-level responses

$$F(X) = \frac{f(\vec{x}_1) \circ f(\vec{x}_2) \circ \cdots \circ f(\vec{x}_N)}{Z}, \tag{1}$$

where $\circ$ denotes the aggregation operator, specific to each MIC algorithm (see below for a review of common operators), and Z represents an optional normalization factor such as $Z = N$ (i.e., dividing the score by the number of instances) or $Z = 1$ if there is no normalization.

The methods falling in this category must address the question of how to infer an instance-level classifier $f(\vec{x})$ without having access to a training set of labelled instances. In order to solve this issue, some assumption must be made about the relationship between the labels of the bags in the training set and the labels of the instances contained in these bags. In this sense, two sub-categories of IS methods emerge clearly in the literature: the ones following the Standard MI (SMI) assumption and the ones following the Collective assumption.[3]

4.1. IS methods following the SMI assumption

The SMI assumption states that every positive bag contains *at least one* positive instance (i.e. an instance belonging to some target positive class), while in every negative bag *all of the instances* are negative. This is an asymmetrical assumption which is used in many MIC problems such as the traditional one of drug discovery in [13]. Note that this assumption is that one of the instances has some desirable properties that *make the bag positive*. Therefore, the methods following this assumption try to identify the type of instance that makes the bag positive.

One of the traditional methods in this category is the Axis-Parallel Rectangle (APR) [13]. In this method, the objective is to estimate an instance-level classifier $f(\vec{x}; \mathcal{R})$ defined as:

$$f(\vec{x}; \mathcal{R}) = \begin{cases} 1 & \text{if } \vec{x} \in \mathcal{R}, \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where R describes an Axis-Parallel Rectangle in the instance space. The parameter R is optimized by maximizing the number of positive bags in the training set that contain at least one instance in R and, at the same time, the number of negative bags that do not contain any instance in R. Based on this, the bag-level classifier can be expressed by using the max rule:

$$F(X) = \max_{\vec{x} \in X} f(\vec{x}) \tag{3}$$

i.e., X is considered positive if at least one of the instances $\vec{x} \in X$ is positive. The max rule is one of the possible aggregation rules used by the different IS methods. In particular, the max rule is also used in DD [15], EM-DD [16] and MI-SVM [17], among other methods. Note that in case of having a binary instance-level classifier $f(\vec{x}) \in \{0, 1\}$, the logical-or aggregation rule

$$F(X) = f(\vec{x}_1) \vee f(\vec{x}_2) \vee \cdots \vee f(\vec{x}_N)$$

is equivalent to the max aggregation rule in Eq. (3). However, if we have a real-valued classifier $f(\vec{x})$, the max rule permits to obtain a real-valued bag-level score $F(X)$ which might be beneficial for some applications.

An algorithm similar to APR is the one based on Diverse Density (DD) [15]. In this algorithm, the instance-level classifier maximizes a DD measure which is high for those points in the instance space that are close to at least one instance of each positive bag and far away from all the instances of negative ones. We refer to [18], and in [19] we provide additional notes that compare this algorithm with APR.

In the MI-SVM method [17], the authors propose an IS classifier $f(\vec{x}; \Theta)$, where Θ are parameters learned by SVM. In order to estimate the SVM, they propose an iterative EM-like approach. In the Expectation-Maximization Diverse Density (EM-DD) algorithm [16], the authors propose a similar iterative approach, maximizing the DD measure in this case. In [19] we provide more details on these algorithms from the point of view of the IS paradigm.

In [20], Bunescu and Mooney propose a Sparse MIL (SMIL) algorithm also based on SVM. The instance-level classifier $f(\vec{x}; \Theta)$ is learned by using a training set of positive of instances $\mathcal{T} = \mathcal{T}^+ \cup \mathcal{T}^-$ defined as follows:

$$\mathcal{T}^+ = \{\mu(X): X \in \mathcal{B}^+\},$$
$$\mathcal{T}^- = \{\vec{x}: \vec{x} \in X \in \mathcal{B}^-\},$$

where $\mathcal{T}^+$ and $\mathcal{T}^-$ are the sets of instances considered positive and the one of instances considered negative, $\mu(X)$ denotes the average of instances inside X, and $\mathcal{B}^+$ and $\mathcal{B}^-$ are the sets of positive and negative bags respectively.

Given this training set, the idea of SMIL is to learn an SVM classifier with a relaxed constraint on the classification of positive instances in $\mathcal{T}^+$. The objective is to avoid forcing the SVM to provide a positive value for all the instances of a positive bag, but only to *at least* one of the instances. For this purpose, the algorithm estimates the parameters Θ of

[3] According to [10] some methods from the BS and ES paradigms follow other assumptions in addition to the mentioned ones, we refer to [10] for a review according to this criteria.

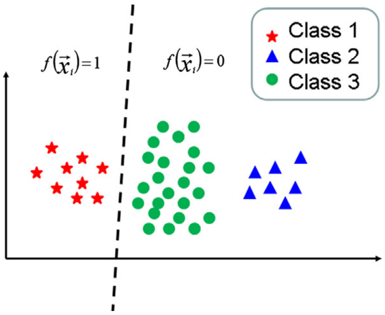


Fig. 8. Illustration of the type of solution that is obtained for SMI-based IS methods (a). See text.

the SVM function $f(\vec{x};\Theta)$ by minimizing a standard SVM objective function (see [20] for details) subject to the following constraints:

$$f(\vec{x};\Theta) \leqslant -1 + \xi_-, \quad \forall\vec{x} \in \mathcal{T}^-, \tag{*}$$

$$f\big(\mu(X);\Theta\big) \geqslant \left(\frac{2}{|X|} - 1\right) - \xi_+, \quad \forall X \in \mathcal{B}^+. \tag{**}$$

The first set of constraints (*) forces the SVM function to provide a negative value when applied to negative instances (allowing a certain degree of misclassification through the slack variable ξ_-). The second set of constraints (**) provides a more relaxed condition for positive instances. This condition depends on the size of the bag X from where $\mu(X)$ is extracted: if X only contains one instance, we have the standard condition $f(\mu(X);\Theta) \geqslant 1 - \xi_+$, i.e., we require the SVM to provide a positive value (allowing again some misclassification). However, if the bag X contains many instances, the threshold imposed on the SVM is gradually more and more relaxed.

In [20], the same authors propose a second IS algorithm named Sparse balanced MIL (SbMIL), which is obtained in two steps: first an SMIL algorithm is trained on the MI data, and then the resulting instance-level classifier $f(\vec{x})$ is applied for labelling the instances of the positive bags. For this purpose, the top ν instances with highest score are labelled as positive and the rest as negative, where ν is a parameter estimated by cross-validation. After this step, a standard SVM classifier is trained using the resulting training set of instances, obtaining the final classifier $f(\vec{x})$. We refer to [20] for other SVM-based IS classifiers, being SbMIL the one with highest performance according to reported results [20].

4.1.1. *Synthetic examples*

The IS methods are not successful when applied to MI data such as the one illustrated in Fig. 5. Here, we need a bag-level discriminative classifier that considers information about the whole bag before taking its decision. Therefore, all of the IS methods will have a poor performance in these situations. This includes the methods discussed below in Sections 4.2 and 4.2.1.

In contrast, IS methods will be successful in the example shown in Fig. 1(a), where positive and negative bags have different types of instances. In Fig. 1(b) we showed a typical decision boundary obtained by methods following the Collective assumption, which we review below. In the case of SMI-based methods, however, the decision boundary obtained is similar to the one shown in Fig. 8. If we compare Figs. 1(b) and 8, the latter has a more asymmetrical division of the space, where the positive region is more adapted to the few instances that appear *only* in positive bags.

4.2. *IS methods following the collective assumption*

The methods of Section 4.1 follow the Standard MI assumption, which has roots in MIC problems such as the Musk drug classification explained in last section, where certain instances make the bag positive. Note that this does not mean that the rest of the instances do not provide relevant information about the bag. For example, in the Musk problem it might happen that all of the instances in a positive bag have certain properties that are characteristic of positive bags. Based on this fact, a more accurate approach is to exploit all the information in order to take a decision. However, the methods from the last section tend to discard a big part of the information, by either only modelling the characteristics of certain instances (as in MI-SVM [17], where only one instance per positive bag is considered in the learning stage, see also our technical report [19]), or by considering only the average vector of a positive bag (as in SMIL [20]).

In this section we present IS methods that make use of the so-called Collective assumption. This assumption states that "*all instances in a bag contribute* ***equally*** *to the bag's label*" [21]. Whereas the SMI assumption considers only a few instances per positive bag, the Collective one considers all of the instances. As discussed above, this type of approach can provide good results in many MIC databases, including the Musk database. In this database, there might be a few instances that are especially relevant, but all the instances inside the bag have characteristics that convey information about the fact that the bag is positive. In general, we find that something similar happens in virtually all the MIC databases.

In order to estimate the instance-level classifier $f(\vec{x})$, the methods of this category use a training set of instances where each instance inherits the label of the bag where it lies. The simplest approach is the SIL algorithm described by Bunescu and Mooney [20], which simply trains a standard supervised classifier $f(\vec{x})$ on the resulting training set. Given a new bag X, the bag-level classifier $F(X)$ is obtained by using the sum as aggregation rule:

$$F(X) = \frac{1}{|X|} \sum_{\vec{x} \in X} f(\vec{x}). \tag{4}$$

Xu et al. [21] and Frank and Xu [22] proposed several methods along these lines. In this work, we evaluate the Wrapper MI method [22], which is simple and representative of this sub-paradigm. The idea of the method is to build a training set using the inheritance rule explained before. In addition to this, the instances are weighted so that each bag receives the same total weight. This is achieved if each instance $\vec{x} \in X$ receives the weight $w(\vec{x}) = \frac{S}{|X|}$, where S is a constant. In [22] the authors argue that this weighting is fundamental to obtain good results, as it makes the different bags of the training set have the same total weight.

4.2.1. *Weighted Collective methods*

A generalization of the previous approach is to allow a different weight for each instance. This generalization gives rise to the weighted Collective assumption, as identified in [10]. Both Foulds [9] and Mangasarian and Wild [23] follow this type of approach. In particular, Foulds proposes an *Iterative Framework for Learning Instance Weights* (IFLIW) which is based on the Wrapper MI algorithm explained before. We refer to [9] and our technical report [19] for more details on this algorithm. Once the weights $w(\vec{x})$ are obtained for each instance $\vec{x}$, the bag classifier $F(X)$ is computed as a weighted sum of instance-level responses:

$$F(X) = \frac{1}{\sum_{\vec{x} \in X} w(\vec{x})} \sum_{\vec{x} \in X} w(\vec{x}) f(\vec{x}). \tag{5}$$

In addition to the weighted Collective paradigm, Foulds and Frank [10] identify what they call the *weighted linear threshold paradigm*. As explained in [10], this paradigm is almost the same as the weighted Collective one, and just a bit more general. In practice, however, the only one algorithm that the authors found to implement this new paradigm is the YARDS method [9] proposed by the same authors. The YARDS algorithm is indeed a Vocabulary-based algorithm, as we will see in Section 7. Thus, we do not introduce the weighted linear threshold paradigm in this work, and instead we describe the YARDS algorithm in Section 7.

As explained in Section 4.1.1, the IS methods do not deal well in situations where the discriminative classifier should consider information beyond the single instance. This type of situations require either BS or ES method, which we review in Sections 5 and 6 respectively.

5. Bag-space paradigm

The idea of the IS paradigm just reviewed is to estimate a model that summarizes the properties of the single *instances*, by discriminating those typically found in positive bags versus those found in negative ones. This makes this type of methods consider *local information*, in the sense that the obtained model is about instances and not about bags as a whole. At classification time, the classifier $F(X)$ is obtained as an aggregation of local responses $f(\vec{x})$, where each of them consider only one instance $\vec{x}$ at a time.

In contrast, the methods of the BS paradigm treat the bags X as a whole, and the discriminant learning process is performed in the space of bags. This allows the algorithm to take into account more information while performing the inference of $F(X)$.

In order to learn a non-vectorial entity such as a bag, we can define a distance function $D(X, Y)$ that compares any two bags X and Y, and plug this distance function into a standard distance-based classifier such as K-NN or SVM (see Section 5.1 for details).

Note that a bag X is nothing else than a set of points in a d-dimensional space. Therefore, any distance function $D(X, Y)$ that compares two sets of points X and Y can be used in this context. In this work we study the minimal Hausdorff distance used in [24], the Earth Movers Distance (EMD) [25], the Chamfer distance [26], and the kernel by Gartner et al. [14]. Let us first see the definition of these functions and in Section 5.2 we discuss the intuition behind them. The minimal Hausdorff distance is defined as:

$$D(X, Y) = \min_{\vec{x} \in X, \vec{y} \in Y} \|\vec{x} - \vec{y}\|. \tag{6}$$

This is the distance between the closest points of X and Y. The EMD distance, on the other hand, is the result of an optimization process. Let $X = \{\vec{x}_1, \dots, \vec{x}_N\}$, and $Y = \{\vec{y}_1, \dots, \vec{y}_M\}$. The EMD distance is defined as:

$$D(X, Y) = \frac{\sum_i \sum_j w_{ij} \|\vec{x}_i - \vec{y}_j\|}{\sum_i \sum_j w_{ij}}, \tag{7}$$

where the weights w_{ij} are obtained through an optimization process that globally minimizes $D(X,Y)$ subject to some constraints, see [27] for details. The Chamfer distance is defined as:

$$D(X,Y) = \frac{1}{|X|}\sum_{\vec{x}\in X}\min_{\vec{y}\in Y}\|\vec{x}-\vec{y}\| + \frac{1}{|Y|}\sum_{\vec{y}\in Y}\min_{\vec{x}\in X}\|\vec{x}-\vec{y}\|. \tag{8}$$

In addition to distance functions $D(X,Y)$, we can use *kernel* functions $K(X,Y)$ that provide a degree of the *similarity* between the sets X and Y. In particular, Gartner et al. [14] propose, among others, the following kernel:

$$K(X,Y) = \sum_{\vec{x}\in X,\vec{y}\in Y} k(\vec{x},\vec{y})^p, \tag{9}$$

where $k(\vec{x},\vec{y})$ is an instance-level kernel and p is in theory related with the size of the largest possible bag, but in practice can be obtained by cross-validation. Usual definitions of the instance-level kernel $k(\vec{x},\vec{y})$ such as the linear, polynomial or Gaussian one, can be seen to provide a measure of similarity or correlation between the instances $\vec{x}$ and $\vec{y}$, so that the bag-level kernel $K(X,Y)$ is the sum of the similarity between instances in X and those in Y. In [14] the authors prove that if the instances $\vec{x}$ are separable in the space induced by the instance-level kernel, then the bags are also separable in the space induced by the bag-level kernel defined in Eq. (9), as long as each positive bag contains *at least one* positive instance, i.e., accomplishes the SMI assumption.[4] Below we explain the intuitive idea under both this kernel and the distances reviewed above.

Along the same line, Zhou et al. [28] propose another kernel function $K(X,Y)$. This kernel not only uses the similarity between pairs $(\vec{x},\vec{y})$ where $\vec{x}\in X$ and $\vec{y}\in Y$, but also uses the similarity between the neighborhood of $\vec{x}$ in X and the neighborhood of $\vec{y}$ in Y, see [28] for the definition. Again, although the authors use this kernel function with SVM, the same kernel function can be used with K-NN as well. The resulting algorithm is called MI-Graph by the authors.

5.1. Distance-based and kernel-based classifiers

The mentioned distance functions can be used with both distance-based classifiers such as K-NN and kernel-based classifiers such as SVM. In case of using an SVM classifier, the distance $D(X,Y)$ can be converted into a kernel $K(X,Y)$ by using the extended Gaussian kernel [29] $K(X,Y)=\exp(-\gamma D(X,Y))$, where γ is a scale parameter estimated by cross-validation.

Conversely, the kernel functions provide some measure of similarity between bags, and they can be used in distance-based classifiers by using the following transformation: $D(X,Y)=\sqrt{K(X,X)-2K(X,Y)+K(Y,Y)}$, as explained in [14].

In addition to K-NN and SVM, in [24] the authors propose a so-called "Citation K-NN" classifier. This classifier is a small modification to the classical K-NN and it can be used in general (not only for MIC problems).

In the results section we show results mostly with the SVM classifier, which is usually more accurate. However, we also evaluate different combinations of classifier and distance functions. In general, the definition of the distance function has a bigger impact in the robustness of the method. In this sense, certain types of distance functions are better at exploiting the information contained in the whole bag and thus make the classifier be more robust in general, no matter if we use SVM or K-NN. Indeed, by defining a distance function $D(X,Y)$ (thus defining an associated kernel $K(X,Y)$) we are providing an *implicit* transformation ϕ from the original bag space to a certain vector space where the bags are described, similar to the ES paradigm that we will see below.

Foulds and Frank [10] identify two separate paradigms: the "Nearest Neighbor" (NN) and the MI-Graph, see Fig. 7. The NN paradigm contains algorithms such as the Citation-KNN, whereas the MI-Graph paradigm only contains the MI-Graph algorithm. As we have seen, our BS paradigm embraces both NN and MI-Graph as special cases and also embraces several other cases in addition to these two, thus being much more general.

5.2. Synthetic examples

Let us see how the different distance functions compare two bags, using as synthetic example the one in Fig. 5. As we discussed in Section 3.3, this figure illustrates the case when global, bag-level information is fundamental for obtaining a good classification of the bags. So it is interesting to see if the different distance functions exploit the global information about the bag by studying how they behave with this example.

Let us first consider the Chamfer distance in Eq. (8). In order to discuss this distance, let us define the distance $d(\vec{x},Y)$, between an instance $\vec{x}\in X$ and a bag Y, as $d(\vec{x},Y)=\min_{\vec{y}\in Y}\|\vec{x}-\vec{y}\|$. This distance will be low if there is some instance in Y that belongs to a class that is similar to the one of $\vec{x}$. Given this definition, we can rewrite the Chamfer distance as $D(X,Y)=\frac{1}{|X|}\sum_{\vec{x}\in X}d(\vec{x},Y)+\frac{1}{|Y|}\sum_{\vec{y}\in Y}d(\vec{y},X)$. Thus, the distance $D(X,Y)$ will be low if two bags X and Y have the same or similar classes of instances.

[4] Note that the fact that the bags are separable does not guarantee that an SVM will find an accurate hyperplane that separates test bags with low error. In the experimental section we present an analysis of this fact.

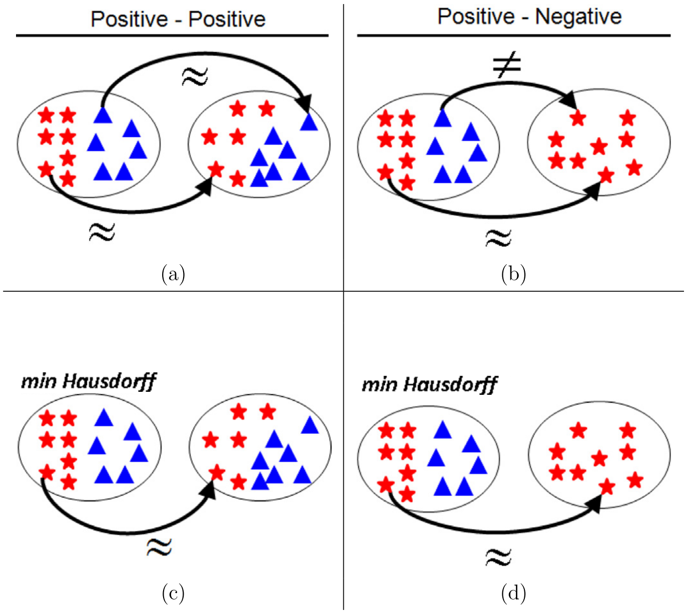


Fig. 9. Matching illustration in case of Chamfer, EMD and Gartner et al. [14] kernel (a) and (b). Illustration for min Hausdorff (c) and (d). See text for explanation.

In Fig. 9(a) this idea is illustrated when comparing two positive bags X and Y. In this case, the distance $D(X,Y)$ is low because each instance of class red star in X matches well with some instance of class red star in Y, and the same happens with blue triangles. Fig. 9(b) illustrates what happens when a positive bag X and negative bag Y are compared. In this case, the distance $D(X,Y)$ will be large because there are many instances that do not match well (in particular, blue triangles in X do not match well with any instance in Y).

A similar thing happens with the EMD and Gartner et al. [14] methods. In the former, an optimization is performed that matches each instance from X with the most similar from Y, in such a way that the global distance between both bags is minimized. In the Gartner et al. [14] method (Eq. (9)), let us consider the instance-level kernel $k(\vec{x},\vec{y}) = \exp(\gamma \|\vec{x} - \vec{y}\|)$, where γ is obtained by cross-validation.[5] In this case, only those instances $\vec{x}$ and $\vec{y}$ that are similar will receive a value $k(\vec{x},\vec{y})$ significantly larger than zero, if γ is correctly estimated. The effect again is that two bags X and Y will receive a high *similarity* score $K(X,Y)$ if the proportion of instances from each class is similar in both bags.

An exception to these methods is the min Hausdorff distance (Eq. (6)), which only considers one matching: the one from the two closest instances in both bags. This is illustrated in Fig. 9(d). In this example, the distance between a positive and negative bag will be low, as there is at least one of the instances from one bag that match well with an instance from the other bag. In general, the min Hausdorff distance is problematic in many situations, as we only extract the information of a single best matching instance, thus missing a lot of information from the rest of the bag. In the experimental section we evaluate the effect of the different distance functions.

6. Embedded-Space paradigm

Both the last paradigm and the one presented in this section are based on extracting global information about the bag. In the BS paradigm this is done in an implicit way through the definition of the distance function $D(X,Y)$ or kernel function $K(X,Y)$. This function defines how bags are compared, and therefore, how the information about them is considered in the matching. v In the ES paradigm, this is done in an explicit way, by defining a mapping $\mathcal{M}: X \mapsto \vec{v}$ from the bag X to a feature vector $\vec{v}$ which summarizes the characteristics of the whole bag. Different definitions of this mapping function put emphasis on different types of information, and have a high impact on the performance of the method.

In this sense, we can split the existing ES methods in roughly two sub-categories. In the first one the methods simply aggregate the statistics of all the instances inside the bag, without making any type of differentiation among instances. In contrast, in the Vocabulary-based paradigm the mapping is constructed by analyzing how the instances of the bag match certain prototypes that have been previously discovered in the data. Let us analyze each of these two sub-paradigms in turn. The first one has only a few methods and is described in Section 6.1. The second one contains a very large number of methods and is described in a separate Section 7.

[5] The parameter γ also subsumes the constant p of Eq. (9) as explained in [14].

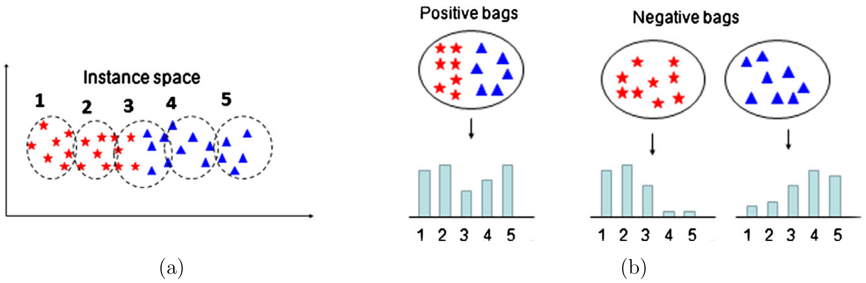


Fig. 10. Illustration of how Bag-of-Words works. See text.

6.1. ES methods without vocabularies

These methods simply aggregate statistics about the attributes of all the instances contained in the bag, without making any differentiation among these instances. Dong et al. [11], for example, propose the so-called Simple MI method that maps each bag X to the average of the instances inside $\mathcal{M}(X) = \frac{1}{|X|}\sum_{\vec{x}\in X}\vec{x}$.

This simple strategy is also evaluated by Bunescu and Mooney [20]. In [14], the authors propose to map each bag to a max–min vector, i.e., $\mathcal{M}(X) = (a_1, \ldots, a_d, b_1, \ldots, b_d)$, where $a_j = \min_{\vec{x}\in X} x_j$ and $b_j = \max_{\vec{x}\in X} x_j$, for $j = 1, \ldots, d$, where d is the dimensionality of the instances. In this work we include the Simple MI method of [11] in our evaluation.

Let us consider the behavior of Simple MI in the scenario depicted in Fig. 5. In this case, the average of positive and negative bags is different, so that the method will be successful. However, when the number of classes of instances is large, using a simple average to describe all of the instances leads to poor performance. In this sense, we will find many cases where the average of two bags is similar, even though each one of them contains different classes of instances. This is evaluated in Section 9.3.

7. Vocabulary-based methods

The methods of this paradigm also use an ES, like the ones of Section 6.1. The difference here is that the instances of the bag are classified (at least in some sense) in order to obtain the embedding. This classification discriminates between different classes of instances, which is not done in the sub-paradigm of Section 6.1. Note that, although we talk about classes of instances, these classes are usually discovered in an unsupervised way, so that they do not have an associated semantic label.

We call this family of methods “Vocabulary-based paradigm” because they make use of a so-called vocabulary in order to perform the embedding. This vocabulary stores the information about all the classes of instances present in the training set, and this information is used in order to first classify the instances of a new bag and then perform the embedding of this bag. All the methods of this family follow *exactly* the steps described in Section 7.2. Before explaining these steps, however, let us explain the idea behind this family of methods.

7.1. Idea behind the vocabulary-based methods

The idea of this paradigm is to provide information about what classes of instances are present in X. In order to clarify the concepts, let us consider the Bag-of-Words (BoW) method, that belongs to this family. Here, the classes of instances are obtained by clustering. The vocabulary V stores the description of the K clusters of instances present in the data. Based on this, a bag X is represented by a histogram $\vec{v}$ that counts how many instances from X fall into each cluster. In this way, the mapping provides information about the composition of X in terms of classes of instances present in X.

Now, let us consider how the BoW method works, using as synthetic example the one in Fig. 5. As we discussed in Section 3.3, this figure illustrates the case when global, bag-level information is fundamental for obtaining a good classification of the bags. So it is interesting to see how the method works in this case. In Fig. 5 the positive bags are characterized by having two classes of instances *co-occurring* in the bag, while negative bags are characterized by having only one of the two classes, *but not both of them* at the same time. In Section 3.3 we showed a real MIC problem where this happens.

Fig. 10(a) shows the clusters discovered in the data. In order to make it more realistic, we have considered that the clusters do not correspond strictly to classes of instances. Instead, the instances of each class are partitioned into two clusters, and there is an additional cluster that contains instances from both classes. Fig. 10(b) shows the mapping of bags into histograms. These histograms reflect the fact that positive bags are characterized by containing instances of both classes in similar proportions, while negative bags only contain one class of instance (i.e., in the latter case only certain components of the histogram will have a large value). These histogram vectors are used by the discriminative classifier in order to discriminate between positive and negative bags. Furthermore, it is able to disregard those regions of the instance space which contain ambiguous information. This is the case of cluster 3, which makes component 3 of the histograms have similar values for both positive and negative bags.

The rest of the Vocabulary-based methods share a similar philosophy, although not all of them are based on clustering. In general, the vocabulary stores a collection of prototypes that are used for describing the composition of the bags. In this sense, all the methods use the vocabulary for describing the content of the bag in terms of the classes of instances found inside. Although the vocabulary always stores prototypes (described in a more or less complex form), in next section we use the more general term “concept” in order to describe the items of the vocabulary.

7.2. Characterization of the Vocabulary-based methods

All the Vocabulary-based methods are based on the following components:

1. A “**vocabulary**” V which is defined as a set $V = \{(C_1, \theta_1), \ldots, (C_K, \theta_K)\}$ storing K “concepts”, where the j-th concept has the identifier C_j and is described by the set of parameters θ_j. Most of the times, the term “concept” means “class of instances”, so that the vocabulary V stores K classes, where the j-th class has identifier C_j and is described by parameters θ_j such as the mean and covariance of the j-th class of instances. Usually each class of instances corresponds to a cluster obtained by K-Means.
2. A **mapping function** $\mathcal{M}(X, V) = \vec{v}$ which, given a bag X and a vocabulary V, obtains a K-dimensional feature vector $\vec{v} = (v_1, \ldots, v_K)$. This provides an embedding of the original bag space into a K-dimensional feature space where each bag X is represented by a feature vector $\vec{v}$. In order to perform this embedding, the function $\mathcal{M}(X, V)$ takes into account the matching between the instances $\vec{x}_i \in X$ and the “concepts” $C_j \in V$. In many cases, this matching can be understood as a classification of instances, i.e., if an instance $\vec{x}_i \in X$ matches the concept $C_j \in V$, then we say that $\vec{x}_i$ is classified as class C_j.
3. A **standard supervised classifier** $\mathcal{G}(\vec{v}) \in [0, 1]$ which classifies the feature vectors $\vec{v}$ in the embedded space. This classifier is trained using an embedded training set $\mathcal{T}_{\mathcal{M}} = \{(\vec{v}_1, y_1), \ldots, (\vec{v}_N, y_N)\}$, where $\vec{v}_i = \mathcal{M}(X_i, V)$, and recall that $y_i \in \{0, 1\}$ is the label of X_i. Given a new bag X to be classified, the bag classifier $F(X)$ can be expressed as $F(X) = \mathcal{G}(\mathcal{M}(X, V))$, i.e., we first map the bag X into the feature vector $\vec{v} = \mathcal{M}(X, V)$, and then apply the classifier $\mathcal{G}(\vec{v})$.

The algorithms of this family differ in the first two components, i.e., how the vocabulary V and the mapping function $\mathcal{M}$ are defined. Below we explain this for each algorithm of the Vocabulary-based family. The third component, the supervised classifier $\mathcal{G}$, is not so important, as we can use any standard classifier such as AdaBoost or SVM.

7.3. Histogram-based methods

The methods of this sub-paradigm use a function $\mathcal{M}$ that maps each bag X into a histogram $\vec{v} = (v_1, \ldots, v_K)$ where the j-th bin v_j counts how many instances of X fall into the j-th class C_j of the vocabulary V. Let us explain how each point of the list in Section 7.2 is instantiated in this sub-paradigm.

The first point of Section 7.2 is the vocabulary V. Here, the “concepts” of the vocabulary are classes of instances. These classes are usually obtained by means of a clustering algorithm, which receives as input the collection of instances of the training set $\mathcal{T}$, and produces as output K classes $C_1, \ldots, C_K$.

The second point of Section 7.2 is the mapping function $\mathcal{M}$. This function can be expressed as follows: $\mathcal{M}(X, V) = (v_1, \ldots, v_K)$, where

$$v_j = \frac{1}{Z} \sum_{\vec{x}_i \in X} f_j(\vec{x}_i), \quad j = 1, \ldots, K. \tag{10}$$

Here, $f_j(\vec{x}_i) \in [0, 1]$ provides the likelihood that the instance $\vec{x}_i \in X$ belongs to class C_j. Thus, v_j counts how many instances are classified into C_j. The constant Z is a normalization factor so that $\sum_j v_j = 1$. We can also set $Z = 1$ and leave the histogram un-normalized.

We see now representative algorithms of the Histogram-based sub-paradigm.

7.3.1. Histogram-based Bag-of-Words with hard-assignment

This algorithm uses hard-assignment (i.e., each instance is assigned to exactly one cluster) in both the clustering algorithm and in the instance classifier $f_j(.)$ of Eq. (10). Let us see both components in turn.

In order to obtain the vocabulary V, we use a clustering algorithm with hard-assignment. Well known examples of such algorithms are K-Means (KM) and Mean-Shift. In this work we use KM, as it is the most widely used [5,30]. Let $\mathcal{D}$ be the set gathering all the instances of all the bags of the training set $\mathcal{T}$. The clustering algorithm receives as input the set of instances $\mathcal{D}$ and produces as output K classes $C_1, \ldots, C_K$, where each instance in $\mathcal{D}$ is assigned to one class. Let S_j be the set of instances in $\mathcal{D}$ assigned to class C_j, and let $\vec{p}_j$ be the average of the instances in S_j, i.e., $\vec{p}_j = \frac{1}{|S_j|} \sum_{\vec{x}_i \in S_j} \vec{x}_i$. The vector $\vec{p}_j$ is called the “prototype” of C_j, and we use it as the parameter that describes C_j in the vocabulary V, i.e., using the notation of Section 7.2 we define $\theta_j = \{\vec{p}_j\}$. The number K of clusters is a parameter of the algorithm, and its choice is discussed in Section 9.2.1.

Regarding the mapping function $\mathcal{M}$ expressed in Eq. (10), we define the instance classifier f_j as:

$$f_j(\vec{x}) = \begin{cases} 1 & \text{if } j = \arg\min_{k=1,\dots,K} \|\vec{x} - \vec{p}_k\|, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

This way, we use hard-assignment i.e., each instance $\vec{x}$ is assigned to exactly one cluster, which is the one with lowest distance to $\vec{x}$. The constant Z in Eq. (10) is set to $Z = |X|$, i.e., the number of instances of the bag, in order to normalize the histogram $\vec{v}$.

This algorithm has been extensively used in Computer Vision under the name of Bag-of-Words, see for example [5,30]. Despite its success it is not well known by the MIC community, and its relationship with other Vocabulary-based algorithms has not been discussed until now. In this work, we call it Histogram-based Bag-of-Words (H-BoW), in order to differentiate it from the Distance-based Bag-of-Words (D-BoW) which will be seen later.

In [4] the authors propose a similar algorithm which uses soft-assignment based on a Gaussian kernel. We refer to [4] and to our technical report [19], Section 1.1.1.

7.3.2. *Bag-of-Words with Gaussian Mixture Models*

Instead of using a clustering algorithm with hard-assignment, as in the last section, we can use a clustering algorithm with soft-assignment. This can be done if we estimate Gaussian Mixture Models with Expectation-Maximization. The resulting algorithm is very similar to the one of the last section, we refer to our technical report [19] for a more detailed discussion. We call the resulting algorithm H-BoW (EM).

7.3.3. *YARDS algorithm*

In [9] the authors propose the "Yet Another Radial Distance-based Similarity measure" (YARDS) algorithm. They claim that this algorithm implements a weighted linear threshold paradigm which in turn generalizes the weighted Collective paradigm, we refer to [9] and [10]. We see here that the YARDS method follows the characterization of Section 7.2 and thus belongs to the Vocabulary-based family according to our analysis. In particular, we classify it in the Histogram-based sub-paradigm, as the mapping function is expressed as in Eq. (10). We see now each point listed in Section 7.2.

The vocabulary V is similar to the one of H-BoW, i.e., the j-th concept C_j is represented by one prototype vector $\vec{p}_j$. In order to obtain the prototypes, we might use a clustering function such as KM, as in the H-BoW algorithm. However, in [9] the authors use all the instances from all the training bags as prototypes. This is like using clusters of size one.

The mapping function $\mathcal{M}$ is expressed again by Eq. (10), where now the function f_j is expressed as $f_j(\vec{x}) = \exp(-\frac{\|\vec{x}-\vec{p}_j\|^2}{\sigma^2})$. Although f_j cannot be strictly considered a classification function, it can be seen as the un-normalized likelihood that $\vec{x}$ falls in a Gaussian with center $\vec{p}_j$ and scale σ. Alternatively, $f_j(\vec{x})$ can be seen as the similarity between the instance $\vec{x}$ and the j-th prototype $\vec{p}_j$ of the vocabulary. The constant Z in Eq. (10) is now set to 1 so that $\vec{v}$ is left un-normalized.

7.3.4. *Weidmann's hierarchy*

This hierarchy consists of three assumptions in increasing order of generality: the presence-based, the threshold-based and the count-based assumptions (see Fig. 7). All of them make use of a set of *target* classes of instances $\mathcal{C} = \{C_1, \dots, C_M\}$. Briefly, the first assumption states that a bag X is positive if, for each $C_i \in \mathcal{C}$ the bag X contains *at least one* instance in C_i. The threshold-based assumption is more general and states that X is positive if it contains at least t_i instances in C_i. And the count-based assumption is the most general one and states that X must contain more than t_i instances and less than u_i instances in C_i, for each $i = 1, \dots, M$.

Let us see in more detail the presence-based assumption, which is the less general of the hierarchy. This assumption was used by Foulds and Frank [10] in order to establish a link between the methods that follow the SMI assumption (see Section 4.1) with methods that, according to our analysis, fall in the Vocabulary-based paradigm. Using the notation of [10], let $\Delta(X, C_j)$ be a function that counts the number of instances in the bag X that belong to class C_j. The presence-based assumption states that a bag X is positive if $\Delta(X, C_j) > 1$ $\forall j = 1, \dots, M$, i.e., if there is at least one instance belonging to each one of the classes in $\mathcal{C}$. Note that Δ is nothing else than a histogram as the one used in the H-BoW method, and that, therefore, the H-BoW method can be considered to follow the presence-based assumption. For this purpose, the H-BoW histogram must be binarized, as it is only important to know the presence or absence of instances in each cluster.

In [10], Foulds and Frank discuss the relationship between the mentioned presence-based assumption and the methods of the IS paradigm that follow the SMI assumption (explained in Section 4.1). The latter methods classify the instances into two classes $\mathcal{C} = \{C_+, C_-\}$, where C_+ denotes the set of positive instances and C_- the set of negative instances. Given such classes and the counting function Δ, we can map the bag X into a histogram with two entries, $\vec{v} = (v_1, v_2)$, where $v_1 = \Delta(X, C_+)$ and $v_2 = \Delta(X, C_-)$, counting the positive and negative instances respectively. The IS methods classify the bag X as positive if any of its instances belongs to C_+, so that we can express the bag-level classifier as $F(X) = [\Delta(X, C_+) > 0]$, where [.] denotes the indicator function.

With this reasoning, Foulds and Frank [10] consider that the IS methods following the SMI assumption are part of the presence-based paradigm. This is consistent with their criteria for categorizing the methods, which is based upon the assumption followed by each method. In our analysis, however, we use a different criteria for categorizing the methods. We look at what level of information the *discriminant* learning takes place: at the instance-level or at the bag-level.

In this sense, we can note two things. First, in the IS methods, the *discriminative* learning takes place at the instance level. This is done for inferring the instance classifier $f(\vec{x})$, which is learned in a discriminative way. This classifier $f(\vec{x})$ is in turn used by the IS methods for classifying the instances into either C_+ or C_-. In contrast, in the ES methods there is no discriminative learning at the instance level. Instead, the clustering of instances $\vec{x}$ is performed in an unsupervised way.

Second, and more important: in the ES methods the *discriminative* learning takes place at the bag level. For this purpose, the ES methods map all the bags X_i of the training set to vectors $\vec{v}_i$. These vectors are then introduced to a standard supervised learner. By feeding these vectors $\vec{v}_i$ into the supervised learner, the ES methods learn bag-level information (because each vector summarizes the content of a whole bag). In contrast, the IS methods do not learn bag-level information. We can see this because the expression $F(X) = [\Delta(X, C_+) > 0]$ is a fixed threshold over the first component of the vector: $F(X) = [v_1 > 0]$ and it does not involve learning the features of $\vec{v}$.

A different thing would be that the whole vector $\vec{v} = (v_1, v_2)$ is passed into a learner. Based on this information, given for all the bags of the training set, the learner could obtain the optimal thresholds for each component v_1 and v_2 in order to obtain the bag-level classifier $F(X)$. This learning process would consider bag-level information represented by $\vec{v} = (v_1, v_2)$. Note that this happens in the ES methods, but not in the IS methods.

As a conclusion, we can see that the SMI *assumption* can be considered a particular case of the *assumption* followed by part of the Vocabulary-based methods (the one based on histograms, which form just a small subset). However, according to the criteria followed in this work (i.e., at what level the information is learned in a discriminative manner) the IS methods in Section 4.1 cannot be considered part of the ES methods using vocabularies.

7.4. Distance-based methods

The previous Histogram-based sub-paradigm is characterized by having a mapping function $\mathcal{M}$ expressed by Eq. (10). This equation *counts* the number of instances that fall into class C_j or that lie close to the prototype $\vec{p}_j$ that represents C_j. We explain now the distance-based sub-paradigm, where the mapping function is expressed as $\mathcal{M}(X, V) = (v_1, \ldots, v_K)$, where:

$$v_j = \min_{\vec{x}_i \in X} d_j(\vec{x}_i), \quad j = 1, \ldots, K. \tag{12}$$

The function $d_j(\vec{x}_i)$ measures the distance between the instance $\vec{x}_i \in X$ and the j-th concept $C_j \in V$. If the concept C_j is represented by just one prototype vector $\vec{p}_j$, we can define $d_j(\vec{x}_i)$ as the Euclidean distance, $d_j(\vec{x}_i) = \|\vec{x}_i - \vec{p}_j\|$.

Note that, by using Eq. (12), the j-th element of the vector $\vec{v}$ indicates the matching degree between the j-th concept C_j and the instances of the bag X. If v_j is low, we can say that C_j has a good matching with some instance of X. On the other hand, if v_j is high, all the instances in X are far away from C_j, which means that C_j does not have a good matching in X.

Therefore, both the Histogram-based methods and the distance-based methods map the bag X to a vector $\vec{v}$ where the j-th element v_j measures the degree of “presence” of the class C_j in the bag X. In the Histogram-based methods this is done by counting the number of instances that fall into C_j, while in the distance-based methods this is done by providing the lowest distance from C_j to any instance in X.

Instead of using a distance function d_j, some algorithms use a similarity function s_j. In this case, instead of using Eq. (12), we use an analog expression:

$$v_j = \max_{\vec{x}_i \in X} s_j(\vec{x}_i) \quad j = 1, \ldots, K. \tag{13}$$

We see now representative algorithms of the distance-based sub-paradigm.

7.4.1. Distance-based Bag-of-Words

Several authors [31–33,1,2] use this algorithm, although this fact is not pointed out in the mentioned papers. Among these methods, probably the most well-known are the one of Auer et al. [2], the DD-SVM method [33], and the MILES method [1]. We first explain our setting of the Distance-based Bag-of-Words (D-BoW) algorithm and then review the setting of other authors [31–33,1,2].

We obtain the vocabulary V as in the H-BoW algorithm: we use clustering with hard-assignment, and each cluster C_j is represented by a single prototype vector $\vec{p}_j$ which is the mean of the instances in the cluster. In our case, we use the KM clustering algorithm as in H-BoW.

Regarding the mapping function $\mathcal{M}$, we tested two approaches: using Eq. (12) with the Euclidean distance, i.e., $d_j(\vec{x}) = \|\vec{x} - \vec{p}_j\|$, and using Eq. (13) where the similarity function s_j is defined as

$$s_j(\vec{x}) = \exp\left(-\frac{\|\vec{x} - \vec{p}_j\|^2}{\sigma^2}\right). \tag{14}$$

We observed slightly better results when we use a similarity s_j instead of a distance d_j, as the values of $s_j(\vec{x})$ are constrained to the interval $[0, 1]$ and rapidly saturate to 0 for far away instances. Therefore, this is the approach reported in the experimental section.

Regarding the methods in the literature, in [33,1,2,32] the vocabulary V is obtained without clustering: the raw instances of the positive bags are used as prototypes $\vec{p}_j$ of the vocabulary. In our experiments we saw that clustering produces better results than using the raw instances of the training set. In [31], the authors use a clustering algorithm with hard-assignment that is specific of their domain. In all the cases, the j-th concept C_j is described by one prototype $\vec{p}_j$, except for [31], where the covariance of the cluster Σ_j is also stored along with the mean of the cluster $\vec{p}_j$.

Regarding the mapping function, in [31] the authors use Eq. (12) where d_j is the Mahalanobis distance: $d_j(\vec{x}) = (\vec{x} - \vec{p}_j)^T \Sigma_j^{-1}(\vec{x} - \vec{p}_j)$. In [2], d_j is the Euclidean distance $d_j(\vec{x}) = \|\vec{x} - \vec{p}_j\|$, and in [33] it is the weighted Euclidean distance $d_j(\vec{x}) = \|\vec{x} - \vec{p}_j\|_{\vec{w}}$. Finally, in [1,32], the authors use Eq. (13) where the similarity function s_j is defined as in Eq. (14), and σ is chosen heuristically.

Regarding the standard classifier $\mathcal{G}$, in [31,2] the authors use AdaBoost with decision stumps, and in [33,1,32] they use SVM. In our additional notes [19] we show how the Auer and Ortner's method [2] can be expressed as a D-BoW algorithm, although it is easy to see it from the original paper [2].

In [10], the authors put the Auer and Ortner's method [2] into the Standard MI paradigm, and the DD-SVM [33] and MILES [1] methods into an isolated paradigm (see Fig. 7). As we have seen, these methods are Vocabulary-based and have the characteristics of the distance-based sub-paradigm (Eqs. (12) and (13)), and in particular we label them as D-BoW methods.

7.4.2. GMIL and count-based GMIL

The Generalized Multiple Instance Learning (GMIL) algorithm [34] explicitly enumerates all possible axis-parallel boxes. It maps the bag X into a boolean vector where the j-th element is set to 1 if at least one instance from the bag falls into the j-th box. It can be easily seen that this method falls into the distance-based sub-paradigm, we refer to our additional notes [19] for details.

7.5. Attribute-based methods

Currently, this sub-paradigm includes only the Intermediate Matching Kernel (IMK) algorithm [6]. In the previous methods, the mapping function $\mathcal{M}$ obtains a vector $\vec{v}$ where the j-th element indicates the level of presence of the j-th concept C_j in the bag X. In the attribute-based sub-paradigm, this is different. Here, the mapping function returns a vector $\vec{v}$ that is a concatenation of sub-vectors:

$$\mathcal{M}(X, V) = \vec{v}_1 \circ \vec{v}_2 \circ \cdots \circ \vec{v}_K \tag{15}$$

where the sub-vector $\vec{v}_j$ summarizes the attributes of the instances in X that match the j-th concept C_j. Note that this is similar to the Simple MI method, but there we only had one vector $\vec{v}$ that summarized the attributes of all the instances of the bag, regardless of their class. In contrast, here we separate the instances of X in classes and then $\vec{v}_j$ summarizes the attributes of the instances inside the class C_j. In general, let the function $f_j(\vec{x}_i) \in [0, 1]$ provide the matching degree between instance $\vec{x}_i \in X$ and concept C_j. The vector $\vec{v}_j$ can be computed as:

$$\vec{v}_j = \frac{\sum_{\vec{x}_i \in X} \vec{x}_i f_j(\vec{x}_i)}{\sum_{\vec{x}_i \in X} f_j(\vec{x}_i)}. \tag{16}$$

For example, f_j might be an instance classifier with either hard-assignment or soft-assignment. This way, the vector $\vec{v}_j$ is the weighted mean of the instances according to its degree of membership in the class C_j. Let us now see the instantiation of these ideas in the IMK method.

In the IMK method, we obtain a vocabulary V as in H-BoW: K-Means is used to obtain K clusters, and for each one we store its center $\vec{p}_j$. Then, the mapping function $\mathcal{M}$ is obtained by defining f_j in Eq. (16) as:

$$f_j(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{x}_i = \arg\min_{\vec{z} \in X} \|\vec{z} - \vec{p}_j\|, \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

Thus, in the IMK method the vector $\vec{v}_j$ is the instance from X that best matches the concept C_j.

Regarding the standard classifier $\mathcal{G}$, in [6] the authors propose SVM with a kernel that is specific for this method. Let two bags X and Y be mapped to vectors $\vec{v}$ and $\vec{w}$ respectively, where $\vec{v} = \vec{v}_1 \circ \cdots \circ \vec{v}_K$ and $\vec{w} = \vec{w}_1 \circ \cdots \circ \vec{w}_K$. We define the kernel $K(\vec{v}, \vec{w})$ as:

$$K(\vec{v}, \vec{w}) = \sum_{j=1}^{K} \exp\left(-\frac{\|\vec{v}_j - \vec{w}_j\|^2}{2\sigma^2}\right).$$

Table 1
List of implemented methods, 'Fam'. indicates the family. All the ES methods are Vocabulary-based, except for Simple MI.

Acronym	Fam.	Section	Acronym	Fam.	Section
SbMIL	IS	3.1	Gartner + SVM	BS	4
EM-DD	IS	3.1	Chamfer + SVM	BS	4
MI-SVM	IS	3.1	Simple MI	ES	5.1
Wrapper MI	IS	3.2	H-BoW (KM)	ES	6.3.1
m.Hauss. + c.KNN	BS	4	H-BoW (EM)	ES	6.3.2
m.Hauss. + SVM	BS	4	D-BoW	ES	6.4.1
EMD + SVM	BS	4	IMK	ES	6.5

7.6. *Methods based on vocabularies of bags*

Currently, this sub-paradigm includes just the BARTMIP method [35]. In this sub-paradigm, the vocabulary V is formed with K concepts C_j where each one represents a class of bags, instead of a class of instances. In order to obtain the vocabulary, the bags of the training set $\mathcal{T}$ are partitioned into K clusters, using a clustering algorithm. For example, in [35] this clustering is performed by using the K-medoids algorithm together with a distance function such as the Hausdorff distance for comparing pairs of bags. The result is that the bags are clustered into K clusters $\mathcal{R}_1, \dots, \mathcal{R}_K$. The j-th cluster $\mathcal{R}_j$ represents the j-th concept C_j of the vocabulary V. In [35], C_j is parameterized by using the medoid of $\mathcal{R}_j$, which is the bag from $\mathcal{R}_j$ with minimum average distance to the other bags of the cluster. Let P_j be this bag medoid.

Regarding the mapping function $\mathcal{M}$, [35] proposes $\mathcal{M}(X, V) = (v_1, \dots, v_K)$ where $v_j = D(X, P_j)$ is the Hausdorff distance between the bag X and the bag medoid P_j.

8. Alternative multi-instance scenarios

In this paper we have focused strictly on the Multiple Instance Classification problem. As we explained in Section 3, the objective of MIC is to classify bags, where each bag is a collection of instances. Let us see here other multi-instance problems. A few authors [36,37] have addressed the problem of instance classification given a multi-instance setting, i.e., when the labels of the training set are assigned to bags, and not to individual instances. We must note that this problem is different from the one of our survey, and therefore the conclusions obtained with our experimental analysis do not necessarily hold for this other problem.

Other multi-instance problems include multi-instance regression [38,35,39], multi-instance multi-label learning [40] and multi-instance clustering [35,41]. The first one is very similar to MIC, where the main difference is that the label associated to the bags is a real value instead of a discrete one, and thus the objective is to learn a real-valued function instead of a discrete-valued classifier. As analyzed in [10], the types of solutions can be categorized in a similar manner to the one employed for MIC (including instance-level solutions using the standard MI assumption [38], embedded-space solutions [35] and bag-space solutions [39]). However, again, the problem is different from MIC and requires its own specific analysis. Another problem is the multi-instance multi-label learning. This is an extension of MIC where each bag can receive several labels, i.e., it can belong to several classes at the same time. Zhou and Zhang [40] extensively analyze this problem and propose several solutions, including some that transform it into a series of MIC problems, we refer also to [10] for a short review. Finally, in multi-instance clustering the objective is to obtain an unsupervised bag-level classification [35,41]. This type of problem is again out of the scope of our review.

9. Experimental evaluation

9.1. *Databases*

We implemented at least one method for each sub-paradigm described in the previous sections. In Table 1 we can find the acronyms of the implemented methods.

We used eight databases from four different fields: Drug Discovery (DD), Information Retrieval (IR), Audio Analysis (AA) and Computer Vision (CV). Table 2 lists the databases and their characteristics. Many of the databases are standard and well-known: the *Musk1* and *Musk2* databases [42] are the most popular ones in the MIC literature, since the early work of Dietterich et al. [13]. The task in these databases is to classify molecules as positive (Musk) or negative (Non-Musk) (see [13]). *Text1* and *Text2* were introduced in [17] and have also become standard since then. In this case, the task is to classify a series of documents as belonging or not to a predefined category. Each document is represented as a bag of feature vectors based on an analysis of the text in the document (see [17]). *Corel1* was introduced in [1] and has also been used by many authors since then. In this case, the task is to classify images into 20 predefined categories, and each image is represented as a bag of instances. In order to convert the Corel data into binary classification problems, we use the well-known one-against-all strategy. *Corel2* was created by us using exactly the same images and the same categories as *Corel1*, and representing the instances with almost the same type of feature vector. The only difference is that the number of instances per bag is more than two orders of magnitude larger than in the original database. This allowed us to evaluate

Table 2
Databases used in the experiments. The fifth column provides the number of positive bags and the number of negative bags for each database.

Database	Domain	Number of bags	Number of instances class per bag	Bags per class (pos./neg.)	Number of dimensions	Classes
Musk1 [13]	DD	92	5	47/45	166	2
Musk2 [13]	DD	102	65	39/63	166	2
Text1 [17]	IR	400	8	200/200	66 552	2
Text2 [17]	IR	400	8	200/200	66 552	2
Speaker	AA	430	1357	240/190	20	2
Corel1 [1]	CV	2000	4	100/1900	9	20
Corel2	CV	2000	6144	100/1900	6	20

the change in performance of the different methods when the number of instances increases. Finally, *Speaker* was created by us based on the audio database available in the website [43] and introduced in the paper [44]. In this case, the task is to identify the gender of the speaker, using as data an audio recording of a sentence spoken by the person. The audio recording is represented as a bag of feature vectors, using a standard representation in the audio processing community [45].

9.1.1. Synthetic database

We also used synthetic data sets for studying the behavior under controlled conditions. A synthetic data set is generated using instance vectors $\vec{x}_i$ that are randomly generated from two Gaussian mixtures: one for the positive class, and another for the negative one. We chose the Gaussian mixture as the underlying distribution of the instances because it can approximate any other distribution if a sufficiently large number of components is used.

The parameters for generating the data set are the number of instances per bag n, the dimensionality of the instances d, the number of Gaussian components for the positive distribution k_p, and for the negative one k_n. By default, we used $k_p = 8$ and $k_n = 32$, i.e., we consider that the negative distribution is more complex and represents the "rest-of-classes". The default values for the rest of parameters were $n = 32$ and $d = 2$. For each configuration of parameters, we randomly generate ten different data sets and report the average classification hit-rate.

9.2. Experimental set up

For the *Musk1*, *Musk2*, *Text1* and *Text2* databases, we used a ten-fold validation approach, as the majority of authors [1]: each round we take 90% of the data set for training and the remaining 10% for testing, and this is repeated ten times in order to test with all the bags. Furthermore, each ten-fold is repeated ten times (each time with a different random splitting), i.e., with a total of 100 rounds of training and testing. For the *Speaker* and *Corel1* databases we used a two-fold validation approach which was repeated ten times, i.e., with a total of 20 rounds of training and testing. In all the cases the average classification accuracy is reported. Finally, for *Corel2*, we just used one round with 50% of the data in the training set and the remaining 50% in the test set. This last setting is frequently employed for databases of this type, as they have a large number of bags.

In the fifth column of Table 2 we indicate the number of bags in the positive and negative class for each data set. If we want to calculate the number of bags in the test set only, we need to multiply by 0.1 (for example, in the *musk* data set we have approximately 5 positive bags and 5 negative bags in the test set), except for the *speaker*, *Corel1* and *Corel2* data sets where we need to multiply by 0.5 (for example, in the *Corel1* data set we have 50 positive bags and 950 negative bags in the test set).

The *Text1* and *Text2* databases have a very large dimensionality (Table 2), which makes it infeasible to apply methods such as H-BoW with EM (due to its little robustness against high dimensionality) and EM-DD due to its computational cost. In order to reduce the dimensionality of the data, PCA could not be applied directly to the original 66K dimensions, due to memory issues with the covariance matrix. We selected the 3000 dimensions that have higher variance, and then applied PCA to reduce to only 65 dimensions.[6]

Many of the evaluated methods require the use of a standard supervised classifier, such as SVM or AdaBoost, as part of the algorithm. With SVM, we used by default an RBF kernel (also called extended Gaussian kernel [29]): $K(\vec{x}, \vec{y}) = \exp(-\frac{1}{\gamma} D(\vec{x}, \vec{y}))$, where $D(\vec{x}, \vec{y})$ is a distance function between two vectors $\vec{x}$ and $\vec{y}$. By default, the Euclidean distance was used, i.e., $D(\vec{x}, \vec{y}) = \|\vec{x} - \vec{y}\|$. The parameter γ was selected, together with the penalty cost C of SVM, using a 5-fold cross-validation as suggested in [46]. By default, SVM was used with vectors scaled to $[0, 1]$ as suggested in [46]. This default setting was changed for EMD with SVM, where we use the EMD distance function, as explained in Section 5, and without scaling the vectors to $[0, 1]$. In the H-BoW algorithm, that use histogram vectors, the χ^2 distance function is employed without scaling the vectors. Finally, the IMK method used a particular kernel, described in Section 7.5, without scaling the vectors. Regarding AdaBoost, we used the version described in [47] with decision stumps and with a very high number of rounds, $K = 10\,000$, which assures a good performance.

[6] We also tried higher dimensionalities, but the results were not better, so we selected a low dimensionality for computational efficiency.

Table 3
Vocabulary sizes for the different databases.

Database	Musk1	Musk2	Text1	Text2	Speaker	Corel1	Corel2
Voc. size	8–256	64–2048	8–256	8–256	64–2048	64–2048	64–2048

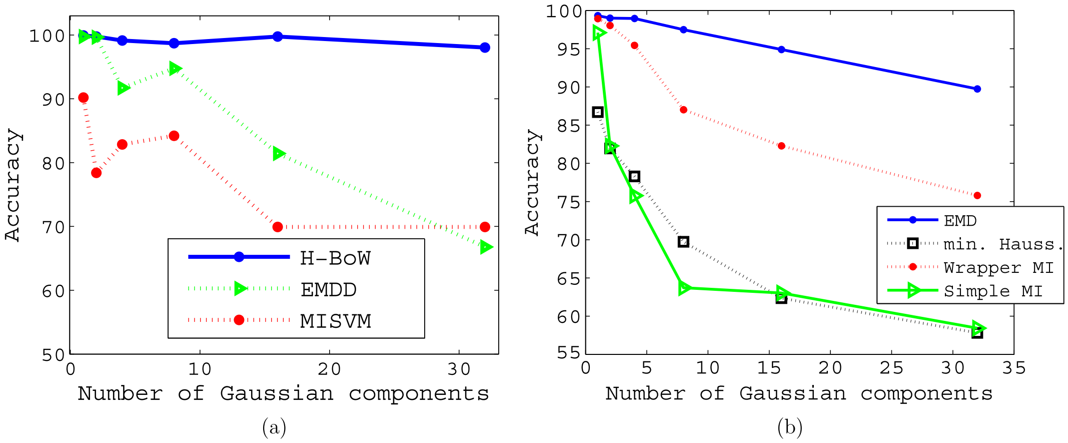


Fig. 11. Performance when bag-level information is necessary. See text.

Finally, regarding the classifier associated with the Vocabulary-based methods, we evaluated both AdaBoost and SVM in all the cases. We heuristically chose SVM as the best performing one for all the methods except for H-BoW (EM), where we chose AdaBoost because it provided slightly better results.

9.2.1. Implementation details for Vocabulary-based methods

In this work we use a common implementation for all the Vocabulary-based methods. We use six different vocabulary sizes $M_1, \ldots, M_6$ which were obtained as follows. First, the maximum vocabulary size M_6 was chosen as half the total number of instances N_{inst} of the database, i.e., $M_6 = \lfloor \frac{N_{inst}}{2} \rfloor$, but without exceeding a maximum size of $M_6 = 2048$ (in case of very large data sets), in order to limit the computational cost. Based on the maximum size M_6, the rest of the sizes were obtained by successively dividing by powers of two: $M_1 = \frac{M_6}{2^5}, \ldots, M_5 = \frac{M_6}{2}$. Table 3 indicates the minimum and maximum vocabulary sizes for each database.

For each vocabulary size $M_i, i = 1, \ldots, 6$, we compute ten vocabularies by using different random initializations in the clustering algorithm. In order to combine all these vocabularies, we saw experimentally that a good approach consists of concatenating the vectors (i.e., the histograms in the case of H-BoW) obtained from all the vocabularies. We tested other settings and the results obtained were similar. The important thing is to use several vocabulary sizes and several initializations. In [19] we include a detailed algorithmic description of our implementation.

Regarding the D-BoW algorithm explained in Section 7.4.1, σ is chosen heuristically as $\sigma = 1$, which produces good results. As explained in Section 7.4.1, we observed slightly better results when we use a similarity s_j instead of a distance d_j, as the values of $s_j(\vec{x})$ are constrained to the interval $[0, 1]$ and rapidly saturate to 0 for far away instances. Therefore, this is the approach reported in the experimental section.

9.3. Impact of using bag-level information: evaluation on synthetic data

Probably the most important difference between paradigms is at what level the *discriminant* information is extracted: at the instance-level, or at the bag-level. In this section we evaluate the performance of the methods in a situation where bag-level information is necessary. In order to do so, we use synthetic data similar to the toy example in Fig. 5, that has been used along the work as a running example. In particular, we evaluate the situation where positive bags have N classes of instances co-occurring at the same time in the bag. As N increases, the information provided by each individual class of instances is less discriminative, and we must consider the combination or co-occurrence of several classes in the bag, which can only be done with bag-level information.

In Fig. 11 we evaluate the performance as a function of N, where we model each class of instances as a Gaussian. Regarding the negative bags, we used a separate set of Gaussian classes of instances. Furthermore, we used a constant high value $M = 32$ for the number of instances. This is done in order to build negative bags with heterogeneous data. This permits simulating a scenario that is common in real problems: while positive objects usually present homogeneous characteristics, negative objects belong to the "rest-of-the-world", i.e., they form a heterogeneous class.

The results in Fig. 11(a)–(b) confirm the following remarks done along the review. First, the IS methods perform poorly, having a high drop in performance as the number of classes N that co-occur increases. This is due to the fact that, in

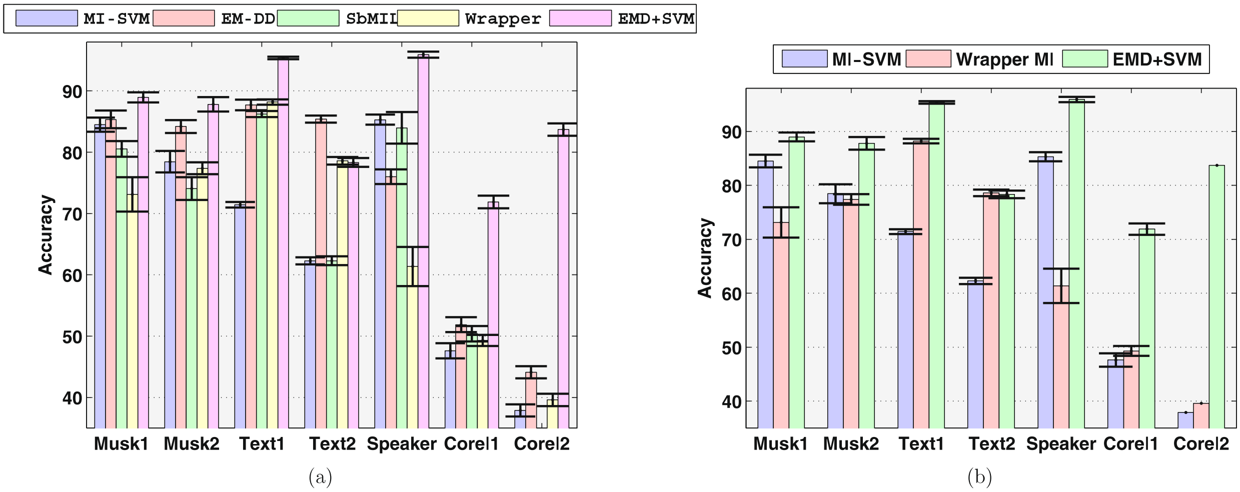


Fig. 12. Performance of IS methods. On top of the bars we show the statistical confidence intervals. The figure is best seen in color. In order to see it in gray-level, we must take into account that the order of the bars corresponds to the order of the methods listed in the legend.

this situation, global bag-level information becomes necessary. Second, the type of assumption, SMI or Collective, does not characterize the performance of the methods in this situation, as all the IS methods (MI-SVM, EM-DD and Wrapper MI) have similar low performance. Third, BS methods perform well as long as the distance function fully exploits information about the whole bag. In this sense, all the distance functions are good except for the min Hausdorff, as we commented in Section 5.2. In Fig. 11(b) we show that the EMD distance performs well while the min Hausdorff does not, and in Section 9.5 we see that all the distances perform well, except for the min Hausdorff.

Fourth, ES methods perform well as long as the embedding conserves the relevant information. In this sense, H-BoW (a Vocabulary-based method) performs well (see Fig. 11(a)), while Simple MI (a method not based on vocabularies) is not robust (see Fig. 11(b)). These observations are confirmed later with real data.

In conclusion, none of the IS methods are robust in a situation where bag-level information is necessary. On the other hand, both the BS and ES paradigms provide an appropriate framework for dealing with this situation. However, the BS paradigm depends on defining a distance function that exploits information from the whole bag, and the ES paradigm depends on defining a mapping that conserves the relevant information of the bag. In this sense, reducing the information to a simple average per bag (as in Simple MI) is not robust.

In the rest of the experimental section we evaluate if these conclusions are confirmed with real data.

9.4. *Performance of IS methods*

In order to see if the results on synthetic data are confirmed on real data we should take into account especially the *Corel1* and *Corel2* databases. These databases define image classification problems similar to the one discussed in Section 3.3 and displayed in Fig. 6. As we saw in that section, in this type of database the bags are characterized by the co-occurrence of several classes of instances, so that using bag-level information becomes fundamental.

Fig. 12 shows the performance of the IS methods using a bar chart.

Fig. 12(a) shows the results of all the IS methods, and also includes the results of the EMD + SVM method. The latter method will be included as a baseline in all the comparisons, as this method provides consistently good results for all the databases. We can see that the performance is dramatically low in *Corel1* and *Corel2* for all the IS methods, as compared with EMD + SVM. As we see later, this behavior is unique to IS methods, i.e., other paradigms do not show this dramatic drop in performance (see Figs. 13(a) and 14(a)). Looking at the results in the rest of the databases, the difference is not so dramatic. But still, *the IS methods do not compare well in the rest of databases either*. Section 9.7 provides ranking results with statistical significance tests, which show that IS methods perform *significantly* worse in general, i.e., taking into account all the databases.

Fig. 12(b) focus on the performance of two IS methods: Wrapper MI, which follows the Collective assumption, and MI-SVM, which follows the SMI assumption. MI-SVM performs better in some databases and Wrapper MI in some others. It does not seem that the SMI assumption makes any clear difference, for example in databases such as Musk2 where it is considered by many authors that the SMI assumption applies well. These conclusions are confirmed by synthetic results as we will see in Section 9.8.

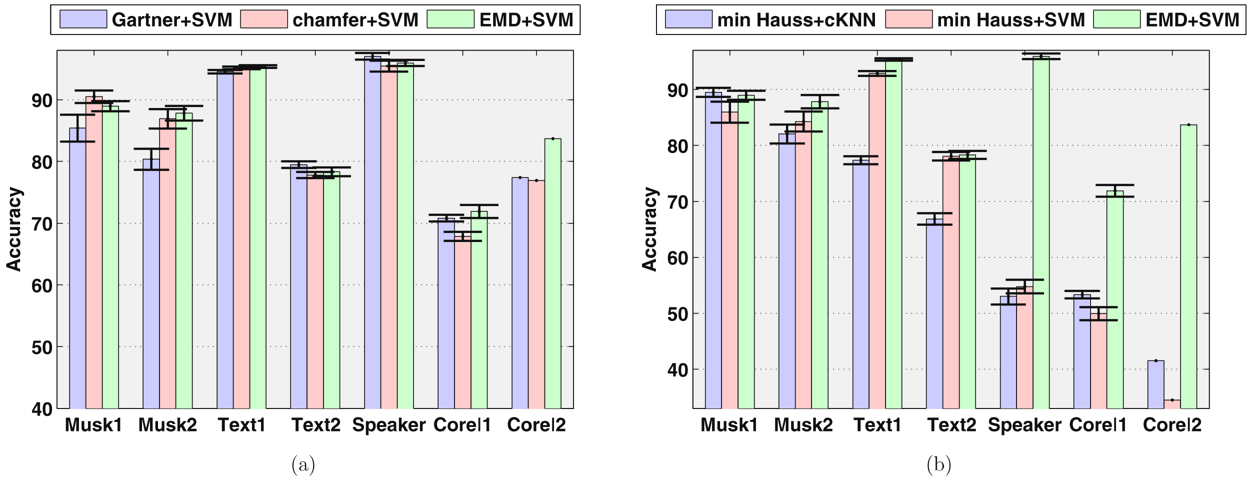


Fig. 13. Performance of BS methods. On top of the bars we show the statistical confidence intervals, except for *Corel2* where only one test was made.

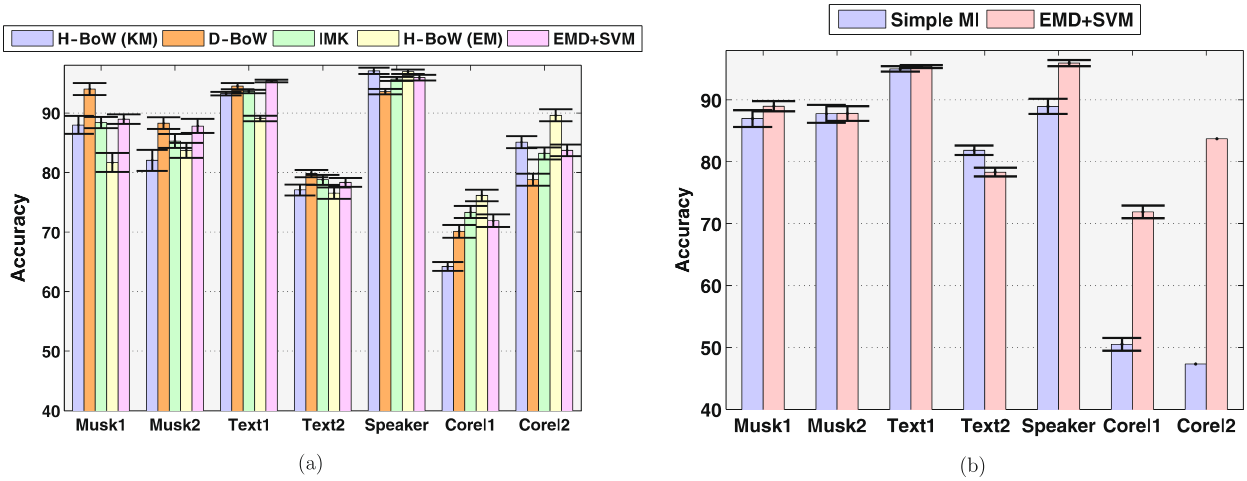


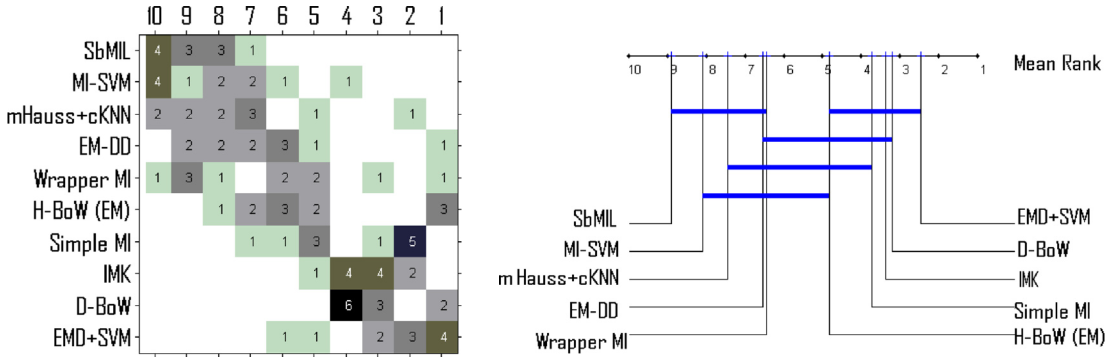
Fig. 14. Performance of ES methods. On top of the bars we show the statistical confidence intervals.

9.5. Performance of BS methods

Fig. 13(a) shows the results of BS methods where the distance function is EMD, Chamfer and the one of Gartner et al. [14]. In all the cases SVM is used. We can see that the performance is comparable and there are no big drops in accuracy for any database. Fig. 13(b) shows the results when the min Hausdorff distance is used. Results for both SVM and Citation-KNN classifiers are shown. We see that the performance drops very significantly for *Corel1* and *Corel2* databases, in both cases. This confirms the results obtained in Section 9.3 under synthetic conditions. In addition to this, we see that min Hausdorff is not appropriate for databases such as Speaker where there is a very large number of instances per bag (see Table 2). The reason is simple: min Hausdorff only considers the matching between the closest two instances, and does not consider whether the rest of the instances are similar or not. Therefore, when we have a large number of instances in the bag we miss a lot of information. The conclusion again is that, while the BS paradigm provides an appropriate framework for dealing with global, bag-level information, the distance function must be defined in an appropriate way.

9.6. Performance of ES methods

Fig. 14(a) shows the results of Vocabulary-based ES methods, including the EMD + SVM for reference purposes. We can see that the performance is comparable and there are no big drops in accuracy for any database. Fig. 14(b) shows the performance of the Simple MI method, which is an ES method not based on vocabularies. We can see again that the performance drops heavily for the Corel1 and Corel2 databases. As in the BS case, the results confirm that, while the ES paradigm provides an appropriate framework for dealing with global, bag-level information, the mapping function must be defined in an appropriate way. In this sense we see that the Vocabulary-based family provides an appropriate framework.



	10	9	8	7	6	5	4	3	2	1
SbMIL	4	3	3	1						
MI-SVM	4	1	2	2	1		1			
mHauss+cKNN	2	2	2	3		1			1	
EM-DD		2	2	2	3	1				1
Wrapper MI	1	3	1		2	2		1		1
H-BoW (EM)			1	2	3	2				3
Simple MI				1	1	3		1	5	
IMK						1	4	4	2	
D-BoW							6	3		2
EMD+SVM					1	1		2	3	4

(a) (b)

Fig. 15. Ranking of methods across all the databases, and statistical significance results. Number of times that each method obtained each rank (a), and critical difference diagram [49].

9.7. Global ranking

In order to summarize the performance on all the databases, we computed the mean rank and computed the Meneyi test for evaluating the statistical significance, following the work of [48] which is in turn based on the recommendations in [49]. As discussed in these papers, very rarely a classification method ranks first or has a constant ranking across different databases, and the only way to statistically test if one method is superior to others is to evaluate the average rank of each method across many databases. In particular [49] recommends to use at least ten databases. Therefore, we added four more databases in order to perform the statistical significance test, but restricted the number of evaluated methods to eight. The new databases are *fox*, *tiger*, *elephant*, and *Text3* which are well-known and were proposed in [17]. Here we focused on the performance of ten methods for computational reasons.

Fig. 15(b) shows the mean rank of the methods evaluated in the eleven databases, and the result of the statistical significance test: those methods linked with a blue bar cannot be said to be statistically different (i.e., there is no sufficient evidence given the number of databases and variability of the performance). However, the test shows that *there is* sufficient statistical evidence that *all of the IS methods* are significantly worse than the EMD + SVM method, as they tend to rank in the lowest positions in all the databases.

9.8. Evaluation of SMI assumption

We also evaluated whether the IS methods based on the SMI assumption are successful in a situation where this assumption holds. Remember that the SMI assumption states that a bag must be classified as positive if and only if it contains *at least one* positive instance. This means that these methods should be able to classify the bags even if they contain a small proportion of positive instances, being the rest of instances negative. This is evaluated in Fig. 16(a). This figure shows the performance of two SMI-based IS methods (MI-SVM and EM-DD) and an ES method (H-BoW). Clearly, the performance of all the methods is very low when the proportion of instances is small, at the level of random chance. We also show the confidence intervals as vertical bars. The intervals are overlapped in the left part of the curve, meaning that difference in performance between methods is not statistically significant when the proportion of relevant instance is small.

Fig. 16(b) evaluates the performance of Gartner et al.'s method [14]. As explained in Section 5, the authors prove that, if the instances are separable in the space induced by the instance-level kernel k, then the resulting bags are separable in the space induced by the bag-level kernel K defined in Eq. (9). To test this, we generated linearly separable instances and we used a linear kernel k in order to obtain the bag kernel K. Then, we tested the performance with varying proportions of positive instances in the positive bag. Fig. 16(b) shows the results, for various choices of the parameter p in Eq. (9). The best performance is obtained with $p = 1$, which is consistent with the paper that suggests $p = 1$ when the instances are completely separable. However, the method has poor performance when the proportion of positive instances is low. This could be due to the size of the training set, which should be very large for obtaining good performance. However, we used a training set with one thousand bags, which is ten times larger than the one needed for perfect classification of instances (we only needed to train with 100 instances), and still the performance was poor. We did not test larger training sets to keep the computational cost reasonably low, and also because, in practice, many real MIC problems are defined with much smaller sets.

Given these results, we can conclude that the performance of the methods does not seem to be affected by the fact that they follow or not the SMI assumption. This has been shown here with synthetic data and also above with real data. In contrast, the methods are clearly more affected by whether the discriminant information is extracted at the instance or at the bag level. In this sense, the IS methods following the SMI assumption perform similarly to other IS methods that do not

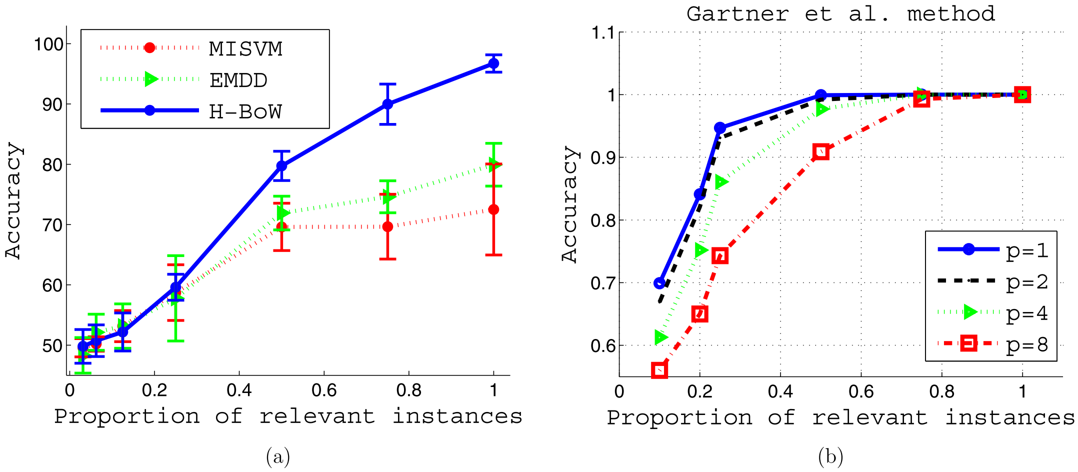


Fig. 16. Performance under SMI assumption, see text.

follow it (and all of them have a low performance in general), while the BS method of Gartner et al. performs similarly to other BS methods.

9.9. Computational cost

Another variable that should be considered when choosing a method is its cost. Let M be the number of bags in the training set, N the average number of instances per bag, and D the dimensionality of the instances. The cost of the BS paradigm is dominated by the computation of the distance function, which is $O(N^2 \times M^2 \times D)$, i.e., for every pair of bags X and Y in the training set we must compute the distance between any instance from X to any other from Y, which has cost $N^2 \times D$. This cost quickly becomes very high when both the number of instances per bag N and the number of bags M is large. This cost is even higher if we use the EMD distance, as the algorithm is based on optimization and has cost $O(N^3 \times M^2 \times D)$ in the worst case. This is problematic, for example in problems such as Content-Based Image Retrieval where there are thousands of instances per bag and thousands of bags. For the *Corel2* database we had to reduce the number of instances per bag by taking only 32 cluster means per bag. The consequence is that the accuracy decreases.

Regarding the ES paradigm, the cost is divided into first computing the vocabulary and then mapping every bag to a single vector. Regarding the latter, the cost is typically $O(M \times N \times K \times D)$, where K is the size of the vocabulary. This is, for every bag X, we compute the distance from every instance in X to every prototype in the vocabulary, which typically involves a distance between two D-dimensional vectors, although it could be higher depending on the particular algorithm. Note that the BS cost is quadratic while the ES one is linear. Regarding the construction of the vocabulary, the cost typically has the same magnitude as the mapping, i.e., $O(n_{iter} \times M \times N \times K \times D)$, for algorithms such as K-Means (similar for GMM-EM if we use diagonal covariance matrices), where n_{iter} is the number of iterations. Other algorithms might have other costs.

Regarding the IS methods, the cost is dominated by the fact that the instance-level classifier is estimated with a very large training set of instances. Note that each bag might contain a large number of instances, so that the training set used in IS methods is much larger than the one of BS or ES methods. In particular the IS methods are not practical when the number of instances per bag is very large (e.g., Speaker or Corel2 databases) and the learning algorithm is SVM. In this case we have to reduce the number of instances by clustering. Apart from these considerations, the cost of the IS algorithms depends also on the optimization algorithm used to identify potential positive instances. This is very dependent on the particular algorithm. In practice, the ES paradigm was significantly less expensive and scaled much better when N or M increases than the other two paradigms.

Another aspect related with the computational cost is the complexity of the learning algorithms. This is discussed in more detail in our technical report [19], but let us explain briefly why the implemented methods have a similar underlying complexity. This is due to the fact that we use the same learning algorithm (SVM with a standard RBF kernel) for almost all the methods, and therefore the learning complexity is similar. In this sense, the difference between the three paradigms (IS, BS and ES) does not lie in the specific learning algorithm, but in the way the information is extracted and introduced to SVM. This is discussed in more detail in our technical report [19].

9.10. Summary of results

The results consistently show the following conclusions. First, using global, bag-level information becomes fundamental in order to obtain accuracy under different MI data. This is confirmed by both synthetic and real data. In this sense, both the BS and ES paradigms present appropriate frameworks for designing MIC methods. In contrast, the IS paradigm provides non-robust methods which have big drops in performance. Overall, statistical tests show that they are significantly worse than the other paradigms.

Second, the classical SMI-based paradigm does not seem to provide a good framework for characterizing methods in terms of performance. In contrast, the fact that a method is IS or BS does have a clear impact on the performance. In this sense, the result shows a similar behavior for all IS methods, regardless of whether they follow the SMI assumption or not. The same happens with the BS methods. This can be seen both in real and synthetic databases.

Third, both BS and ES provide an appropriate framework for exploiting information from the whole bag. However, an appropriate distance function must be defined if we use BS, and an appropriate mapping must be defined if we use ES. In this sense, we have shown that the distance function must evaluate to what extent every instance in one bag has a similar instance in the other bag. The intuition behind this is discussed in Section 7.1. In fact, this idea is followed, in one way or the other, by all the distance functions except for the min Hausdorff. Regarding the ES methods, we have shown that using a Vocabulary provides a better framework than other types of embedding such as Simple MI. The intuition behind the Vocabulary-based paradigm is discussed in Section 7.1.

Fourth, BS methods become infeasible when either the number of instances per bag or the number of bags are large. In this situation we should consider using vocabulary-based algorithms.

10. Conclusions

This work presents the first analysis of the MIC methods that include both a complete review and empirical comparison of the different paradigms. In our analysis, we obtain a compact categorization of the methods based on how they manage the information. The performed categorization pursues two important objectives: first, to provide a clear picture about the existing approaches, and second to allow an experimental analysis of the different categories of methods, in such a way that methods of the same category perform similarly under different situations.

Regarding the empirical study, we provided an extensive analysis that included fourteen algorithms from the three paradigms. In order to test these algorithms, we used seven databases from four different domains of knowledge. We also included a synthetic database where we were able to study the behavior under controlled conditions. Altogether, the results show clearly that using global, bag-level information leads to superior performance. In this sense, both BS and ES methods can be used, but taking into account certain important design considerations: in the BS methods we must consider the similarity between all the instances, and in the ES method using a Vocabulary-based mapping is important. This confirmed the analysis provided in the review section, where these design considerations were justified. Finally, we saw that, under certain conditions, the BS approaches become computationally very expensive and, therefore, using vocabulary-based ES methods is a better choice for certain databases.

Summarizing, in addition to provide a clear picture of the type of existent MIC solutions, we also studied important guidelines for obtaining MIC methods that are robust in different databases. In this sense, we must note that, given a new MIC problem we are faced with, it is difficult to know the type of method that is most suitable for it. Therefore, finding a set of methods that have a consistently good performance in many situations is important. Furthermore, this analysis can also help to improve the design of future algorithms by providing a basic framework. In this sense, it becomes clear in our analysis that the discriminative classifier must be based on global information from the whole bag, and it also becomes clear how this can be done appropriately. Furthermore, we showed that certain design choices such as following the well-known SMI assumption do not seem to have an impact on the performance, and that it has more impact to focus on new effective ways to extract global information in such a way that the interrelations between instances inside the bag are characterized.

Acknowledgements

We would like to thank Dr. Razvan Bunescu for providing the implementation of methods in [20], and Dr. Antonio M. López for valuable feedback. We would also like to thank the anonymous reviewers for their valuable comments. This work was supported by the fellowship RYC-2008-03789 and the spanish project TRA2011-29454-C03-01.

References

[1] Y. Chen, J. Bi, J.Z. Wang, Miles: Multiple-instance learning via embedded instance selection, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 1931–1947.

[2] P. Auer, R. Ortner, A boosting approach to multiple instance learning, in: Proc. of European Conference on Computer Vision, 2004, pp. 63–74.

[3] H. Lee, A. Battle, R. Raina, A.Y. Ng, Efficient sparse coding algorithms, in: Proc. of Neural Information Processing Systems, 2007, pp. 801–808.

[4] J.C. van Gemert, C. Veenman, A. Smeulders, J. Geusebroek, Visual word ambiguity, IEEE Trans. Pattern Anal. Mach. Intell. 32 (7) (2010) 1271–1283.

[5] J. Sivic, A. Zisserman, Video google: A text retrieval approach to object matching in videos, in: Proc. of IEEE Computer Vision and Pattern Recognition, 2003, pp. 1470–1477.

[6] S. Boughorbel, J.-P. Tarel, N. Boujemaa, The intermediate matching kernel for image local features, in: Proc. of International Joint Conference on Neural Networks, 2005, pp. 889–894.

[7] Z.-H. Zhou, M.-L. Zhang, Solving multi-instance problems with classifier ensemble based on constructive clustering, Knowl. Inf. Syst. 11 (2) (2007) 155–170.

[8] N. Weidmann, E. Frank, B. Pfahringer, A two-level learning method for generalized multi-instance problems, in: Proc. of European Conference on Machine Learning, 2003, pp. 468–479.

[9] J. Foulds, Learning instance weights in multi-instance learning, Master's thesis, University of Waikato, 2008.

[10] J. Foulds, E. Frank, A review of multi-instance learning assumptions, Knowl. Eng. Rev. 25 (1) (2010) 1–25.

[11] L. Dong, A comparison of multi-instance learning algorithms, Master's thesis, University of Waikato, 2006.

[12] J. Amores, Vocabulary-based approaches for multiple-instance data: A comparative study, in: Proc. of International Conference on Pattern Recognition, 2010, pp. 4246–4250.

[13] T.G. Dietterich, R.H. Lathrop, T. Lozano-Pérez, Solving the multiple-instance problem with axis-parallel rectangles, Artificial Intelligence 89 (1997) 31–71.

[14] T. Gärtner, P.A. Flach, A. Kowalczyk, A.J. Smola, Multi-instance kernels, in: Proc. of International Conference on Machine Learning, 2002, pp. 179–186.

[15] O. Maron, T. Lozano-Pérez, A framework for multiple-instance learning, in: Proc. of Neural Information Processing Systems, 1998, pp. 570–576.

[16] Q. Zhang, S.A. Goldman, EM-DD: An improved multiple-instance learning technique, in: Proc. of Neural Information Processing Systems, 2001, pp. 1073–1080.

[17] S. Andrews, I. Tsochantaridis, T. Hofmann, Support vector machines for multiple-instance learning, in: Proc. of Neural Information Processing Systems, 2003, pp. 561–568.

[18] M. Oded, Learning from ambiguity, PhD thesis, Massachusetts Institute of Technology, May 1998.

[19] J. Amores, Notes on "Multiple instance classification: Review, taxonomy and comparative study", Tech. rep., Universitat Autònoma de Barcelona, 2012, http://www.cvc.uab.es/~jaume/additionalNotes.pdf.

[20] R. Bunescu, R. Mooney, Multiple instance learning for sparse positive bags, in: Proc. of International Conference on Machine Learning, 2007, pp. 105–112.

[21] X. Xu, Statistical learning in multiple instance problems, Master's thesis, University of Waikato, 2003.

[22] E. Frank, X. Xu, Applying propositional learning algorithms to multi-instance data, Tech. rep., Department of Computer Science, University of Waikato, 2003.

[23] O. Mangasarian, E. Wild, Multiple instance learning via successive linear programming, J. Optim. Theory Appl. 137 (3) (2008) 555–568.

[24] J. Wang, J.-D. Zucker, Solving the multiple-instance problem: A lazy learning approach, in: Proc. of International Conference on Machine Learning, 2000, pp. 1119–1125.

[25] J. Zhang, M. Marszalek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: A comprehensive study, Int. J. Comput. Vis. 73 (2) (2007) 213–238.

[26] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (24) (2002) 509–522.

[27] Y. Rubner, C. Tomasi, L.J. Guibas, The earth mover's distance as a metric for image retrieval, Int. J. Comput. Vis. 40 (2) (2000) 99–121.

[28] Z.-H. Zhou, Y.-Y. Sun, Y.-F. Li, Multi-instance learning by treating instances as non i.i.d. samples, in: Proc. of International Conference on Machine Learning, 2009, pp. 1249–1256.

[29] O. Chapelle, P. Haffner, V. Vapnik, Support vector machines for histogram-based image classification, IEEE Trans. Neural Netw. 10 (5) (1999) 1055–1064.

[30] E. Nowak, F. Jurie, B. Triggs, Sampling strategies for bag-of-features image classification, in: Proc. of European Conference on Computer Vision, 2006, pp. 490–503.

[31] A. Opelt, A. Pinz, M. Fussenegger, P. Auer, Generic object recognition with boosting, IEEE Trans. Pattern Anal. Mach. Intell. 28 (3) (2006) 416–431.

[32] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, T. Poggio, Robust object recognition with cortex-like mechanisms, IEEE Trans. Pattern Anal. Mach. Intell. 29 (3) (2007) 411–426.

[33] Y. Chen, J.Z. Wang, Image categorization by learning and reasoning with regions, J. Mach. Learn. Res. 5 (2004) 913–939.

[34] S. Scott, J. Zhang, J. Brown, On generalized multiple-instance learning, Int. J. Comput. Intell. Appl. 5 (1) (2003) 21–35.

[35] M.-L. Zhang, Z.-H. Zhou, Multi-instance clustering with applications to multi-instance prediction, Appl. Intell. 31 (1) (2009) 47–68.

[36] A. Vezhnevets, J.M. Buhmann, Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning, in: IEEE Proc. of Computer Vision and Pattern Recognition, 2010, pp. 3249–3256.

[37] B. Settles, S. Ray, M. Craven, Multiple-instance active learning, in: Proc. of Neural Information Processing Systems, 2008, pp. 1289–1296.

[38] S. Ray, D. Page, Multiple instance regression, in: Proc. of International Conference on Machine Learning, 2001, pp. 425–432.

[39] R. Amar, D. Dooly, S. Goldman, Q. Zhang, Multiple-instance learning of real-valued data, in: Proc. of International Conference on Machine Learning, 2001, pp. 3–10.

[40] Z.-H. Zhou, M.-L. Zhang, Multi-instance multi-label learning with application to scene classification, in: Proc. of Neural Information Processing Systems, 2006, pp. 1609–1616.

[41] H. Kriegel, A. Pryakhin, M. Schubert, An EM-approach for clustering multi-instance objects, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2006, pp. 139–148.

[42] A. Frank, A. Asuncion, UCI machine learning repository, http://archive.ics.uci.edu/ml, 2010.

[43] C. Sanderson, http://www.itee.uq.edu.au/~conrad/vidtimit/.

[44] C. Sanderson, B.C. Lovell, Multi-region probabilistic histograms for robust and scalable identity inference, in: Lecture Notes in Computer Science, vol. 5558, 2009, pp. 199–208.

[45] D.A. Reynolds, T.F. Quatieri, R.B. Dunn, Speaker verification using adapted Gaussian mixture models, Digital Signal Processing 10 (2000) 19–41.

[46] C.-W. Hsu, C.-C. Chang, C.-J. Lin, A practical guide to support vector classification, Tech. rep., National Taiwan University, 2003.

[47] P. Viola, M.J. Jones, Robust real-time face detection, Int. J. Comput. Vis. 57 (2) (2004) 137–154.

[48] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: An evaluation of the state of the art, IEEE Trans. Pattern Anal. Mach. Intell. 34 (4) (2012) 743–761, http://dx.doi.org/10.1109/TPAMI.2011.155.

[49] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.