An ensemble approach to multi-view multi-instance learning

Alberto Cano[a], José María Luna[b], Sebastián Ventura[b,c]

[a]*Department of Computer Science, Virginia Commonwealth University, USA*
[b]*Department of Computer Science and Numerical Analysis, University of Cordoba, Spain*
[c]*Department of Computer Science, King Abdulaziz University, Saudi Arabia Kingdom*

Abstract

Multi-view learning combines data from multiple heterogeneous sources and employs their complementary information to build more accurate models. Multi-instance learning represents examples as labeled bags containing sets of instances. Data fusion of different multi-instance views cannot be simply concatenated into a single set of features. This paper proposes an ensemble classification approach to conduct a fusion of the views and pursues consensus among the predictions to take advantage of the complementary information from multiple views. Importantly, the ensemble must deal with the different feature spaces coming from each of the views, while data for the bags may be partially represented in the views. The experimental study evaluates and compares the performance of the proposal with 20 traditional, ensemble-based, and multi-view algorithms on a set of 15 multi-instance datasets. Experimental results indicate the better performance of ensemble methods than single-classifiers, but especially the best results of the multi-view multi-instance approaches. Results are validated through multiple non-parametric statistical analysis.

Keywords: Multi-view, multi-instance, classification, ensemble

1. Introduction

Multi-view learning [1] is a relatively new paradigm that exploits data represented by multiple distinct feature sets called views which have been obtained from different data sources. The objective is to learn functions to model the views and jointly optimize and exploit the redundancy and complementarity of data in the views [2, 3]. The multi-view foundations are

based on the consensus and complementary principles of data. The consensus principle maximizes the agreement of distinct views, i.e. the relationships between different subsets of features. The complementary principle advocates for the distribution of the information among the views, i.e., each view contains partial data that may not provide interesting information when analyzed separately, but when merged together provide meaningful and comprehensive knowledge to the learner. Multi-view learning has received much attention in machine learning, especially for multi-label and image classification [4, 5, 6, 7]. Learning classification models from the fusion of multiple views increases the strength of the classification predictions as compared from the predictions on independent views [8]. However, it is not straightforward to accomplish such task and many times multi-view data cannot be easily integrated into learners due to the heterogeneous data representation from the multiple sources.

Multi-instance learning [9, 10, 11, 12] is a generalization of traditional supervised learning in which each observation or object, called *bag*, comprises a bag identifier and a variable number of non-repeated *instances* described by feature vectors. The bag is associated with a single class label, although the labels of the particular instances are unknown. Dietterich et al. [9] defined the multiple instance problem, in which a bag is classified as positive if it contains at least one positive instance. More recently, other generalized multi-instance models have been formalized [13, 14]. The intersection between multi-view and multi-instance learning is natural and represents a flexible representation paradigm for supervised learning. Multi-instance learning can be adapted to train from data allocated in multiple views, reflecting the distribution of the information into heterogeneous feature sets with different sets of disjoint attributes. The fusion of the views provides more complete information about the bags that should lead to better accuracy as compared with the prediction using the partial data from the isolated views.

Despite the flexibility of the multi-instance data representation, it is difficult to combine multiple multi-instance views. While bags contained in multiple views share the bag identifier, the instances are represented using different feature vectors (attributes provided by each view). In traditional single-instance classification, it is straightforward to combine data from two views having common examples by simply joining the feature sets. However, this process cannot be directly performed in multi-instance learning since there is no matching between the particular instances but between the whole bags by means of their bag identifiers. Current multi-view multi-instance

approaches perform problem transformations of the data into multiple single meta-instance views [15]. Nonetheless, this *flattening* transformation destroys the multi-instance representation of the data and refuses to resolve the true multi-view multi-instance nature of the problem.

This paper presents an ensemble approach to directly learn from multi-view multi-instance (MVMI) data. Ensembles combine the predictions of multiple classifiers in order to reduce the variance and bias of the predictions, and specifically they have shown to improve accuracy in single-instance multi-view [16]. Our main contributions are the following:

1. Propose ensemble learning to fuse the information from multi-view multi-instance data relations with heterogeneous feature sets without conducting any multi-instance problem transformation.
2. Identify the best performing base classifier for each of the views. Views are represented by different feature vectors providing diverse information. Therefore, rather than using the same base classifier on all of the views, we propose to evaluate different families of base classifiers on each of the views, and then selecting the model which performed best for each view. Moreover, considering the prediction of a diverse family of classifiers has shown to improve performance [17].
3. Weight the prediction of the base classifiers based on their local accuracy on the views. This is motivated because not all the views are going to provide useful high-quality information to the classifier, but on the contrary, some views may provide irrelevant, contradictory, or noisy information to the ensemble. Therefore, the weighting will disregard predictions coming from low-quality classifiers, in contrast to the default majority voting that would decrease the performance of the ensemble.

The experimental study evaluates and compares the performance of 20 multi-instance classifiers on a set of 15 datasets with regards of five performance measures. The experimental results are validated through the analysis of non-parametric statistical tests [18], namely the Bonferroni–Dunn, Nemenyi, Holm, and Wilcoxon tests that evaluate whether there are statistically significant differences between multiple and pairwise comparisons of algorithms. Results indicate that ensemble methods improve the performance of traditional multi-instance classifiers, while our multi-view multi-instance approach overcomes the results of the multi-view single meta-instance as well.

This paper is structured as follows. Section 2 reviews related works. Section 3 describes the proposed multi-view multi-instance approach. Section 4 describes the experimental study, discusses and analyzes the results. Finally, Section 5 presents the conclusions.

2. Background

This section defines the basis and reviews related works on multi-instance learning, multi-view learning, and ensemble classification.

2.1. Multi-instance learning

In multi-instance classification the examples are called bags, and represent a set of instances. The class is associated with the whole bag although the instances are not explicitly associated with any particular class. Therefore, multi-instance learning inducts a prediction function $f(bag) \rightarrow C$ where the bag is a set of k instances $\{\bar{x}_1, \bar{x}_2, ..., \bar{x}_k\}$ associated with a class label C. Instances are feature vectors $\bar{x} = [x_1, x_2, ..., x_f]$ with f attribute–values.

The standard hypothesis by Dietterich [9] assumes that if a bag is positive, then at least one of its instances must be positive. However, if the bag is negative, then none of its instances could be positive. Therefore, a bag is positive if and only if at least one of its instances is positive. This can be modelled by introducing a second function $g(bag, j)$ that takes a single variant instance j. The classification function $f(bag)$ can be defined as follows:

$$f(bag) = \begin{Bmatrix} 1 & if\ \exists\ j & | & g(bag, j) = 1 \\ 0 & & & otherwise \end{Bmatrix} \tag{1}$$

Foulds and Frank [13] reviewed the basis of the generalized multi-instance learning assumptions. *Amores* [11] presented a recent review describing the taxonomy of algorithms in multi-instance classification. Methods are categorized based on whether they focus on instance-level information (instance-space paradigm) or bag-level information, and in the latter whether they extract the relevant information implicitly (bag-space paradigm) or explicitly (embedded-space paradigm). *Weidmann et al.* [19] presented a two-level method which transformed a multi-instance dataset into single meta-instance dataset. They create multiple propositional single-instance datasets using decision trees and clustering, which are learned using a correlation-based multi-view learner, at the cost of losing the multi-instance representation.

Langone and Suykens [20] combined both instance level and bag level information for aggregated feature learning, the collective assumption is used to express the relationship between the instance labels and the bag labels.

2.2. Multi-view learning

Multi-view learning algorithms can be categorized into three groups: co-training, multiple kernel learning, and subspace learning. Co-training learns from different views alternately to maximize the mutual agreement among two distinct views of the data [21], i.e., the consensus principle. It relies on three assumptions: sufficiency, compatibility, and conditional independence. Multiple kernel learning performs linear or non-linear combinations of the views. Subspace learning focuses on the latent subspace shared by views [22, 23], which has lower dimensionality. Many related works on multi-view multi-instance learning are mainly oriented to image annotation or semi-supervised learning [24, 25, 26, 27, 28].

Wang et al. [15] presented MI-TLC, a two-level method which transforms a multi-view multi-instance dataset into a multi-view single meta-instance dataset. The first level creates multiple single-instance views using decision trees and clustering, which are learned using correlation-based learners. The second level constructs the final classification model using regular propositional classifiers combined in a multi-view algorithm. This is closest related work to our proposal, but they conduct flattening of the multi-instances into single-instances via problem transformation. On the contrary, our contribution handles multi-view multi-instance data directly.

Wu et al. [29] modelled music emotion recognition as multi-label multi-layer multi-instance multi-view problem using a hierarchical structure. *Zhang et al.* [30] incorporated the consistencies between the views into multi-instance learning for document classification. *Wu et al.* [31] formulated a cross-view feature selection to identify the most representative features across the views in multi-instance.

Li et al. [32] proposed a multi-view learning approach called *co-labeling* for ambiguous data in semi-supervised learning, multi-instance learning and max-margin clustering. The classifiers are trained on different views, and they teach each other by iteratively passing the prediction of examples from one classifier to others. To train a classifier with a label candidate set for each view, they adopt the Multiple Kernel Learning (MKL) technique by constructing the base kernel through associating the input kernel calculated from input features with one label candidate.

Nguyen et al. [33] presented a multi-view multi-instance multi-label framework based on hierarchical Bayesian network and variational interference, but this research is focused on semi-supervised learning on images dataset and compares with single-view multi-instance multi-label.

2.3. Ensemble classification

The combination of several base classifiers to build an ensemble is focus of intense research. These systems perform information fusion at different levels overcoming limitations of traditional approaches based on single classifiers. Therefore, they are natural learners for multi-view datasets. *Rokach* [34] presented a review and taxonomy of ensemble classifiers. The flexibility of ensembles has also allowed for their application in multi-instance learning problems [35].

Wózniak et al. [36] presented a survey on multiple classifier systems from the point of view of hybrid intelligent systems. They analyzed the domains, topology, design and diversity of classifiers fusion into ensembles, and their application to real-world problems [37, 38].

Zhou and Zhang [39] presented a constructive clustering ensemble method that groups the instances within the bags into several clusters. Bags are mapped into a boolean feature space where each attribute corresponds to a cluster, and the value of an attribute is set to 1 if and only if that bag has an instance in that cluster. A single-instance model is built on the resulting dataset. The algorithm is repeated for multiple number of clusters, and classification predictions are made via a majority vote of the resulting ensemble of single-instance classifiers.

Kang et al. [40] introduced the concept lattice and ensemble learning into multi-instance learning. They divide the problem into multiple local multi-instance problems based on concept lattice. Then, local target sets are identified in each local problem. Finally, the whole dataset can be classified by an ensemble of the multiple local target feature sets. Lattice-computing ensemble has also been applied to the fusion of disparate data types [41].

Xu et al. [42] presented a classification ensemble for multi-instance multi-label video annotation. The ensemble helped to overcome the class imbalance problem among data classes. This method first samples several subsets from majority class independently, then trains multiple classifiers using the subsets and minority class, and finally combines all the classifiers for a final decision. It is able to exploit the majority class examples ignored by under-sampling, and make the training speed faster due to the small size training set.

Sener and Ikizler-Cinbis [43] worked on the problem of image re-ranking with multi-instance ensembles. They showed that with a simple ensemble of multi-instance classifiers that was only based on visual content of the retrieved images and without using any user feedback, they were able to achieve quite successful re-ranking of the images. They constructed set of candidate bags with extracted features on which the multi-instance ensemble was built.

Lin et al. [16] presented an ensemble learning approach from multiple information sources via label propagation and consensus. They proposed two label propagation methods to infer the labels of training objects from unlabeled sources by making a full use of class label information from labeled sources and internal structure information from unlabeled sources. Then, they predict the labels of testing objects using the ensemble learning model of multiple information sources. The approach considered multiple information sources to provide plentiful and complementary information for the same object set and to perform better than a single source.

Zhang et al. [44] proposed an ensemble manifold regularized sparse low-rank approximation for multi-view feature embedding considering the complementary property among multiple features.

Hong et al. [45] presented an ensemble manifold recognizing method based on multi-view data fusion. Laplacian matrices learned from different views are combined to obtain a local optimal solution that embeds multi-view information iteratively [46].

Karakatic and Podgorelec [47] developed allocation ensembles that separate data instances based on anomaly detection and allocates them to one of the micro classifiers, built with the existing classification algorithms on a subset of training data. The outputs of micro classifiers are then fused together into one final classification.

Majority voting and weighted voting are two commonly used strategies in the combination of the ensemble predictions [48, 49]. Simple majority voting does not consider the quality of each of the base classifiers, while weighted voting allows to increase the weight for high-quality base classifiers and demote it for low-quality ones. This is especially interesting in the case of multi-view learning, since some views may provide high-quality information while others may not, or even introduce noise in the ensemble, and under the presence of imbalanced data [50, 51, 52, 53] which frequently happens in multi-instance learning.

3. Multi-view multi-instance ensemble

This section presents the multi-view multi-instance representation of the information and describes the ensemble approach to build classifiers for multi-view multi-instance datasets.

3.1. Representation of the information

Multi-view learning represents examples with multiple views of feature vectors. A multi-view example x is represented in the form of $\{\bar{x}^1, ..., \bar{x}^V\}$ where $\bar{x}^v$ is the feature vector for the v-th view and V is the number of views. Each view v is described by a feature space of χ^v attributes. Therefore, each $\bar{x}^v = [x_1, x_2, ..., x_f] \epsilon \chi^v$ is a feature vector for the view v with f attributes.

Multi-view multi-instance learning representation extends the multi-view representation but now the examples are bags of instances as defined in Section 2.1. Therefore, a multi-view multi-instance example is formally represented in the form of a bag b with multiple views $\{b^1, ..., b^V\}$ associated with a class label C. Each bag contains for a view v a set of k_v instances and it is defined as $b^v = \{\bar{x}^v_1, \bar{x}^v_2, ..., \bar{x}^v_{k_v}\}$. Each instance is defined on the view feature space as $\bar{x}^v = [x_1, x_2, ..., x_{f_v}] \epsilon \chi^v$.

It is essential to highlight two major advantages of this flexible representation. First, bags may comprise a completely different number of instances for each of the views. Second, a bag is not necessarily present in all of the views, i.e. it is feasible that a view does not contain data for a particular bag identifier. This allows to reflect the behavior of many real-world information systems because many times information is missing when coming from different sources, and only partial information is available for a given example, yet we want the classifier to take full advantage of the partially available information in different views rather than disregard the incomplete data. Therefore, this flexibility allows to handle data with a significant imbalance in the amount of information available within the multiple views, which also happens in real-world systems where some information sources provide much more quantity of data characteristics than others.

For clarification, let us consider an example of a multi-view multi-instance dataset with three views containing two sample bags as shown in Figure 1. The former with BagID Bag-1 contains two, three, and one instances for each view respectively. The latter with BagID Bag-2 contains tree instances for the second view, once instance for the third view, but it does not contain any instance for the first view because no data was provided for that view.

This is different to the concept of missing value for a specific feature that is represented using the question mark "?". A significant advantage of this representation is that it is readable by the Weka software tool by using multiple relational attributes, then running the available multi-instance algorithms.

BagID	View 1			View 2		View 3				Class
	F1	F2	F3	F4	F5	F6	F7	F8	F9	
Bag-1	0.2	0.1	0.4	0.7	0.2	0.9	0.1	0.4	0.5	Positive
	0.3	0.2	0.4	0.5	0.3					
				0.6	0.2					
Bag-2				0.8	0.9	0.2	0.6	0.7	?	Negative
				0.9	0.8					

Figure 1: Example of multi-view multi-instance representation.

3.2. Multi-view multi-instance ensemble

In single-view classification ensembles are built learning multiple base classifiers on a given set of training examples and features. Boosting, bagging, and stacking are popular methods to reduce the bias of particular algorithms and the minimize the variance of the predictions, then increasing the accuracy of the ensemble as compared with individual classifiers. In multi-view classification the base classifiers are employed to learn from each of the views. The consensus principle advocates to maximize the agreement when predicting from different sets of features, in order to improve the accuracy as compared with the predictions from individual views. The complementary principle supports learning from relevant information distributed in the views. We may define formally a multi-view multi-instance ensemble composed of base multi-instance classifiers $f^v(\bar{x}^v)$ build on each view and the ensemble prediction $g(x)$ arises from the fusion of the base classifiers from all views $g(x) = fuse(f^1(\bar{x}^1), ..., f^V(\bar{x}^V))$, e.g. conducted via a voting function.

3.2.1. Identification of the base classifier for each view

A multi-view bagging approach with homogeneous classifiers per view would reduce the variance but limit the diversity of models. A multi-view boosting approach to build an ensemble using a different set of potential classifiers seems reasonable to minimize the ensemble error. The key for the success of the ensemble is the agreement of diverse classifiers [54]. When

analyzing the advantages of random forests or rotation forests [55] it is noticed that the diversity of components trained by *feature bagging* has the effect of averaging out the variance and bias of the diverse collection of trees. Moreover, it is not known which kind of classifier is more appropriate for each view nor how much of the information contained in each view can be trusted, as some attributes may be more relevant for classification while others only introduce noise into data and should be omitted. Therefore, we propose to build an ensemble that first evaluates a set of candidate multi-instance base classifiers from diverse families to determine which works best for each view (based on the idea presented in [24]). Specifically, we considered 3 top performance multi-instance learners available in the Weka software tool to obtain predictions from complementary families of algorithms, including SimpleMI [56], MISMO [57], and TLC [56]. Evaluating the candidate classifiers is conducted in parallel to minimize the performance impact. Eventually, the base learner which obtains the best local accuracy (based on the partial data in a view) is selected as component of the ensemble for that view.

3.2.2. Weighted voting

The reliability of all the base classifiers should not be equally considered because they are built from different data views with different feature vectors and information quality. Data contained in a particular view may be redundant, irrelevant, or even introduce noise to the classifier. Thus, should the predictions from these classifiers were considered as reliable as others (as in bagging or simple majority vote), they would decrease the accuracy of the predictions. Therefore, we consider that it is necessary to introduce a weighted voting to determine the importance of the votes of the base classifiers in the ensemble by means of the local accuracy of each base learner on its view, which was the criterion employed to select the base classifier model. Eventually, this allows to totally disregard predictions from a view if they provide low quality predictions violating the consensus principle.

The algorithm for the multi-view multi-instance ensemble (MVMI) is detailed in Algorithm 1. The base classifiers are selected using an internal 10-fold cross validation within each train fold to determine which candidate learner performs best for each view. Note that the prediction of a base classifier may be either positive, negative, or void, which means that the base learner refrains from giving a prediction because the given test bag does not contain any information associated for a particular view. Finally, the weighted voting is performed among the predictions.

Algorithm 1 MVMI ensemble algorithm

Require: TR = train data, TS = test data, CC = candidate base classifiers

1. Select base classifiers and build ensemble

for every view v in V **do**
  for every candidate classifier C_c in CC **do**
    Run 10-fold CV on view TR^v using candidate classifier C_c
    $acc_c \leftarrow$ accuracy of C_c in TR^v
  end for
  $base_v \leftarrow$ select best $C_k \mid acc_k = max(acc)$
  $weight_v \leftarrow acc_k$
end for

2. Classify test data

for every example *bag* in TS **do**
  for every view v in V **do**
    if *bag* contains data for view v **then**
      $baseClassPrediction \leftarrow classify(base_v, bag^v)$
      $votes[baseClassPrediction] \mathrel{+}= weight_v$
    end if
  end for
  $ensembleClassPrediction \leftarrow argmax(votes)$
end for

4. Experiments

This section presents and discusses the experimental study carried out to evaluate the and compare the performance of traditional, ensemble, and multi-view multi-instance classifiers. First, the experimental setup is described. Second, the results are presented and analyzed. Third, a statistical analysis is conducted to determine statistical differences among the performance of the approaches. Fourth, performance degradation is analyzed with regards of the missing data on views. Finally, the run time of the algorithms is presented and their scalability is discussed.

Table 1: Datasets information.

Dataset	Attributes	Bags	Instances
eastWest	26	20	213
westEast	26	20	213
suramin	22	13	2898
mut-atoms	12	188	1618
mut-bonds	18	188	4081
mut-chains	26	188	5424
trx	10	193	26611
tiger	232	200	1188
fox	232	200	1320
elephant	232	200	1391
musk1	168	92	476
musk2	168	101	6728
component	22	3130	36894
function	202	5242	55536
webmining	5865	113	3423

4.1. Experimental setup

Experiments were run on a set of 15 multi-instance datasets from the Weka [56] and KEEL [58] dataset repositories. Information of the datasets is detailed in Table 1, which comprise a wide variety of sizes as measured by the number of attributes, bags, and instances.

The experimental comparison comprises 20 multi-instance learning algorithms from different families. Traditional and classical methods: CitationKNN [59], MDD and MIDD [60], MIEMDD [61], MILR [56], MIOptimalBall [62], SimpleMI [56], MISVM and MISMO [57], MIRI [56], MINND [63], MITI [64], and TLC [56]. Ensemble-based: Vote [65], AdaBoost and MIBoost [66], and Stacking [67]. Multi-view methods: MI-TLC [15]. All experiments were carried out on a 10-fold cross-validation. The parameter settings for the algorithms were set using the recommend values by the authors of the algorithms. The proposed MVMI method does not require any parameter.

In order to analyze the results from the experiments, non-parametric statistical tests are used [18, 68]. To evaluate whether there are significant differences in the results of the algorithms, the Bonferroni–Dunn, Nemenyi, Holm, and Wilcoxon rank-sum tests are used to find the significant differences occurring between approaches.

Table 2: Accuracy results.

Accuracy	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	0.4500	0.6000	0.6500	0.6600	0.6800	0.7400	0.6100	0.7200	0.5700	0.7200
westEast	0.4986	0.5728	0.3408	0.2639	0.3840	0.2554	0.2751	0.6479	0.3840	0.4836
suramin	0.6364	0.6364	0.6182	0.2545	0.4182	0.6364	0.6364	0.6364	0.6364	0.7818
mut-atoms	0.6830	0.7138	0.7053	0.6734	0.7106	0.6521	0.6649	0.6894	0.7543	0.3426
mut-bonds	0.7106	0.6883	0.7585	0.7234	0.7372	0.6926	0.6649	0.8021	0.7660	0.3298
mut-chains	0.7553	0.7543	0.7809	0.7064	0.7362	0.7021	0.6649	0.8309	0.7904	0.3713
trx	0.8332	0.8705	**0.9026**	0.8922	0.8591	0.8984	0.8705	0.8705	0.7710	0.8176
tiger	0.5000	0.6660	0.7000	0.6890	0.6760	0.6170	0.7440	0.7390	0.7150	0.6530
fox	0.5000	0.6580	0.6010	0.5970	0.5710	0.5610	0.4940	0.5380	0.6070	0.5000
elephant	0.5000	0.7770	0.8190	0.7410	0.7750	0.7300	0.7790	0.8150	0.7880	0.5000
musk1	0.7071	0.7832	0.7739	0.7176	0.6387	0.5319	0.5647	0.7214	0.5752	0.7071
musk2	0.6653	0.6653	0.7188	0.7921	0.6455	0.7683	0.6891	0.6871	0.6594	0.6653
component	0.8649	0.8649	0.8810	0.8705	0.9006	0.8691	0.8675	0.8649	0.9100	0.8649
function	0.9155	0.9155	0.9155	0.9180	0.9380	0.9155	0.9186	0.9155	0.9508	0.9155
webmining	0.8376	0.8376	0.8376	0.8363	0.3730	0.8376	0.6730	0.8376	0.8327	0.8376
Average	0.6705	0.7336	0.7335	0.6890	0.6695	0.6938	0.6744	0.7544	0.7140	0.6327

Accuracy	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.6200	0.6000	**0.9500**	0.7000	0.7000	0.7000	0.5800	0.6000	0.8700	0.9100
westEast	0.2676	0.4751	**0.9249**	0.5390	0.6376	0.5662	0.5174	0.5775	0.7863	0.7991
suramin	0.6364	0.5455	0.5077	0.6364	0.7077	0.7538	**0.8462**	0.7692	0.8065	**0.8462**
mut-atoms	0.7383	0.7766	0.7660	0.7149	**0.8021**	0.8000	0.7660	0.7851	0.7972	0.7851
mut-bonds	0.7691	0.8202	0.7926	0.8266	0.8287	**0.8511**	0.8032	0.8266	0.8226	0.8234
mut-chains	0.7777	0.8606	0.8053	0.8489	0.8128	0.8574	0.8106	**0.8649**	0.8427	0.8596
trx	0.3565	0.8756	0.8705	0.8736	0.8964	0.8943	0.8788	0.8858	0.8634	0.8705
tiger	0.6890	0.6990	0.7110	0.7570	0.7240	**0.7620**	0.7180	0.7520	0.7505	0.7510
fox	0.6260	0.5990	0.6080	0.5900	0.6190	0.6490	0.6470	0.6100	0.6801	**0.6885**
elephant	0.7760	0.8320	0.7750	0.8690	0.8450	0.8530	0.8090	**0.8710**	0.8254	0.8502
musk1	0.5597	0.8634	0.8324	0.7996	0.7160	0.6634	0.8029	0.7861	0.8483	**0.8769**
musk2	0.6337	0.6851	0.6653	0.7366	0.6950	0.7327	0.7030	0.7267	**0.8334**	0.8311
component	0.8939	0.9357	0.9235	0.8810	0.8955	0.9370	0.9127	0.8868	0.9108	**0.9391**
function	0.9473	0.9645	0.9587	**0.9672**	0.9517	0.9506	0.9524	0.9519	0.9514	0.9526
webmining	0.7973	0.8427	0.7818	0.8104	0.8160	0.8332	0.8457	0.8160	0.8541	**0.9042**
Average	0.6726	0.7583	0.7915	0.7700	0.7765	0.7869	0.7729	0.7806	0.8295	**0.8458**

4.2. Results

Table 2 shows the accuracy results of the 20 algorithms (columns) on the 15 multi-instance datasets (rows), together with the average results. The best values are represented in bold-face. Results from base learners are provided first, followed by the ensemble methods, and the last two bottom columns are for the multi-view multi-instance approaches, namely MI-TLC [15] and our method (named MVMI). Results indicate that ensemble methods clearly obtain better results than traditional classifiers. MI-TLC additionally improves the results of ensemble approaches, whereas MVMI achieves the best results. Specifically, our approach achieves the best results in 5 of the 15 datasets, and significantly higher average. Bagging and Stacking are the best among ensemble classifiers.

Table 3: Kappa results.

Kappa	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	-0.1000	0.2000	0.3000	0.3200	0.3600	0.4800	0.2200	0.4400	0.1400	0.4400
westEast	-0.1311	-0.0034	-0.3938	-0.3928	-0.1736	-0.3640	-0.3758	0.2403	-0.2850	0.0133
suramin	0.0000	0.0000	-0.0340	-0.3321	-0.1890	0.0000	0.0000	0.0000	0.0000	0.4710
mut-atoms	0.2158	0.2150	0.2642	0.1509	0.2336	0.1989	0.0000	0.2578	0.4213	-0.0539
mut-bonds	0.3012	0.2254	0.3947	0.2776	0.3897	0.2914	0.0000	0.5608	0.4406	-0.0117
mut-chains	0.4245	0.3915	0.4823	0.3087	0.3801	0.3106	0.0000	0.6167	0.5026	-0.0143
trx	0.0502	0.0000	**0.5070**	0.3882	0.1769	0.4613	0.0000	0.0000	0.1498	-0.0620
tiger	0.0000	0.3320	0.4000	0.3780	0.3520	0.2340	0.4880	0.4780	0.4300	0.3060
fox	0.0000	0.3160	0.2020	0.1940	0.1420	0.1220	-0.0120	0.0760	0.2140	0.0000
elephant	0.0000	0.5540	0.6380	0.4820	0.5500	0.4600	0.5580	0.6300	0.5760	0.0000
musk1	0.4308	0.5590	0.5561	0.4539	0.3014	0.1186	0.1710	0.4525	0.1966	0.4308
musk2	0.3026	0.3026	0.3908	**0.5818**	0.2514	0.4962	0.3358	0.3400	0.2869	0.3026
component	0.0000	0.0000	0.3085	0.1812	0.4637	0.1593	0.0467	0.0000	0.6052	0.0000
function	0.0000	0.0000	0.0000	0.1731	0.4559	0.0000	0.0824	0.0000	0.6794	0.0000
webmining	0.0000	0.0000	0.0000	0.0232	0.0895	0.0000	0.3375	0.0000	0.5015	0.0000
Average	0.0996	0.2061	0.2677	0.2125	0.2522	0.1979	0.1234	0.2728	0.3239	0.1215

Kappa	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.2400	0.2000	**0.9000**	0.4000	0.4000	0.4000	0.1600	0.2000	0.7800	0.8200
westEast	-0.3566	-0.0627	**0.8343**	-0.0046	0.1747	0.0147	-0.0972	0.0832	0.5215	0.5300
suramin	0.0000	-0.1702	-0.0197	0.0000	0.4115	0.5002	**0.6829**	0.5262	0.6509	**0.6829**
mut-atoms	0.3350	0.4826	0.4700	0.2936	**0.5439**	0.5356	0.4467	0.4987	0.4731	0.4859
mut-bonds	0.4127	0.5928	0.5236	0.6062	0.6122	**0.6632**	0.5494	0.6051	0.5888	0.5894
mut-chains	0.4317	0.6862	0.5372	0.6523	0.5806	0.6761	0.5606	**0.6918**	0.6632	0.6765
trx	0.0719	0.3039	0.0112	0.0406	0.4124	0.2946	0.1059	0.2134	0.0023	0.0041
tiger	0.3780	0.3980	0.4220	0.5140	0.4480	**0.5240**	0.4360	0.5040	0.5017	0.5020
fox	0.2520	0.1980	0.2160	0.1800	0.2380	0.2980	0.2940	0.2200	0.3734	**0.3780**
elephant	0.5520	0.6640	0.5500	0.7380	0.6900	0.6932	0.6180	**0.7420**	0.6749	0.6952
musk1	0.1693	0.7239	0.6575	0.6069	0.4465	0.3354	0.6035	0.5787	0.7296	**0.7542**
musk2	0.2669	0.3510	0.3186	0.4343	0.3666	0.4437	0.3711	0.4201	0.2865	0.2879
component	0.5999	0.6926	0.6565	0.2066	0.3662	0.6932	0.5713	0.3817	0.6813	**0.7025**
function	0.6919	0.7510	0.7212	0.7542	0.7195	0.7563	0.6394	0.7182	0.8237	**0.8247**
webmining	0.4565	0.3330	0.2656	-0.0488	0.1109	0.2896	0.3491	0.0026	0.5369	**0.5684**
Average	0.3001	0.4096	0.4709	0.3582	0.4347	0.4745	0.4194	0.4257	0.5525	**0.5668**

Accuracy may be misleading when classes are strongly imbalanced [69, 70], since a default-hypothesis classifier can still achieve a very good accuracy. For instance, the *component* dataset has six times many more negative than positive examples. Therefore, we also report two complementary measures to evaluate the performance of the algorithms considering imbalanced classes. Namely, they are the Cohen's kappa rate [71] and the Area under the curve (AUC) [72]. The Cohen's kappa evaluates the merit of the classifier according to the data class distribution. Kappa statistic ranges from -1 (total disagreement) through 0 (random classification) to 1 (total agreement):

$$Kappa = (N \sum_{i=1}^{k} x_{ii} - \sum_{i=1}^{k} x_{i.} x_{.i}) / (N^2 - \sum_{i=1}^{k} x_{i.} x_{.i}) \quad (2)$$

Table 4: AUC results.

AUC	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	0.3650	0.6220	0.6850	0.7280	0.7450	0.7400	0.6100	0.7200	0.5700	0.7200
westEast	0.2641	0.4958	0.2837	0.2378	0.3460	0.2914	0.2892	0.6181	0.3542	0.5070
suramin	0.0179	0.2214	0.2893	0.1964	0.3714	0.5000	0.5000	0.5000	0.5000	0.7107
mut-atoms	0.6687	0.7393	0.7224	0.5728	0.6810	0.5975	0.5000	0.6207	0.7010	0.4623
mut-bonds	0.6940	0.7333	0.8185	0.6785	0.7964	0.6420	0.5000	0.7835	0.7067	0.4913
mut-chains	0.6929	0.7848	0.8331	0.7247	0.7141	0.6508	0.5000	0.8051	0.7377	0.4902
trx	0.6124	0.8139	0.8251	0.8094	0.7647	0.6931	0.5000	0.5000	0.5961	0.4765
tiger	0.5000	0.7288	0.7428	0.7297	0.6973	0.6170	0.7440	0.7390	0.7150	0.6530
fox	0.5000	0.7027	0.6422	0.6114	0.5551	0.5610	0.4940	0.5380	0.6070	0.5000
elephant	0.5000	0.8473	0.8946	0.8167	0.8289	0.7300	0.7790	0.8150	0.7880	0.5000
musk1	0.8252	0.8853	0.7936	0.7775	0.5953	0.5642	0.5910	0.7337	0.6052	0.8252
musk2	0.6918	0.6918	0.6922	0.8366	0.7073	0.7409	0.6659	0.6700	0.6446	0.6918
component	0.4956	0.4956	0.8225	0.7080	0.8708	0.5531	0.5139	0.4956	0.7952	0.4956
function	0.5504	0.5504	0.5504	0.7697	0.9191	0.5504	0.5236	0.5504	0.8371	0.5504
webmining	0.6144	0.6144	0.6144	0.5072	0.7578	0.6144	0.8048	0.6144	0.8088	0.6144
Average	0.5328	0.6618	0.6807	0.6470	0.6900	0.6031	0.5677	0.6469	0.6644	0.5792

AUC	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.6200	0.6510	0.9050	0.7320	0.7120	0.7720	0.5980	0.5130	0.8991	**0.9540**
westEast	0.2946	0.4216	**0.8103**	0.4533	0.5153	0.4061	0.4812	0.4914	0.7363	0.7483
suramin	0.5000	0.4750	0.4857	0.1321	**0.8381**	0.7214	0.7786	0.6881	0.7602	0.7976
mut-atoms	0.6481	0.7966	0.7541	0.7817	0.8049	0.8326	0.8012	0.8018	**0.8342**	0.8309
mut-bonds	0.6823	0.8613	0.8152	0.8748	0.8701	**0.8884**	0.8381	0.8665	0.8626	0.8681
mut-chains	0.6903	0.8738	0.7960	0.8789	0.8374	**0.8811**	0.8166	0.8717	0.8686	0.8723
trx	0.6099	0.8263	0.4625	0.8783	0.8745	**0.9082**	0.7816	0.8607	0.8679	0.8851
tiger	0.6890	0.7482	0.7139	0.8052	0.7852	0.8052	0.7710	0.7975	0.8098	**0.8181**
fox	0.6260	0.6401	0.6292	0.6334	0.6548	**0.7121**	0.6698	0.6438	0.6701	0.6773
elephant	0.7760	0.8983	0.7854	**0.9364**	0.9180	0.9268	0.8718	0.9351	0.9048	0.9117
musk1	0.5898	0.9103	0.8382	0.9062	0.8268	0.7636	0.8895	0.9015	0.9091	**0.9616**
musk2	0.6398	0.7272	0.6764	0.7608	0.7326	0.7793	0.7115	0.7545	0.8521	**0.8647**
component	0.8399	0.9331	0.8461	0.6371	0.7805	0.9478	0.9282	0.7618	0.9194	**0.9480**
function	0.8817	0.9466	0.8726	**0.9693**	0.8948	0.9682	0.9519	0.8995	0.9518	0.9670
webmining	0.8123	0.8459	0.6492	0.7821	0.7965	0.9115	0.8533	0.8055	0.8919	**0.9442**
Average	0.6600	0.7704	0.7360	0.7441	0.7894	0.8150	0.7828	0.7728	0.8492	**0.8699**

where x_{ii} is the count of cases in the main diagonal of the confusion matrix, N is the number of examples, and $x_{.i}$, $x_{i.}$ are the column and row total counts.

The AUC shows the trade-off between the true positive rate (TPR) and the false positive rate (FPR) [69]. It is calculated as:

$$AUC = \frac{1 + TPR - FPR}{2} \tag{3}$$

Table 3 shows the Cohen's kappa. It is interesting to point out that negative values indicate a performance worse than default-hypothesis. Not a few negative values are included, especially on base learners, and the *westEast* dataset. Similarly, Kappa increases for ensemble methods, and especially for the MVMI approach, which obtains the best results for 6 out of 15 datasets.

Table 5: Precision results.

Precision	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	0.4576	0.6655	0.6639	0.6879	0.6888	0.6841	0.5670	0.7127	0.5663	0.7968
westEast	0.2671	0.3789	0.1403	0.2093	0.3038	0.2268	0.2189	0.5405	0.2165	0.3868
suramin	0.6364	0.6364	0.6291	0.2500	0.5543	0.6364	0.6364	0.6364	0.6364	0.7567
mut-atoms	0.7201	0.7079	0.7307	0.6995	0.7155	0.7281	0.6649	0.7367	0.7883	0.5318
mut-bonds	0.7469	0.7220	0.7626	0.7275	0.7830	0.7555	0.6649	0.8593	0.7880	0.0667
mut-chains	0.7892	0.7648	0.8041	0.7547	0.7774	0.7604	0.6649	0.8651	0.8085	0.6396
trx	0.2030	0.0000	0.6742	0.6716	0.3980	0.6769	0.0000	0.0000	0.2314	0.0315
tiger	0.0000	0.6860	0.7102	0.6681	0.6570	0.6343	0.7276	0.7140	0.7021	0.6901
fox	0.0000	0.6444	0.5976	0.5808	0.5622	0.5658	0.4813	0.5344	0.5861	0.0000
elephant	0.0000	0.7822	0.8123	0.7215	0.7322	0.7372	0.8162	0.7881	0.7246	0.0000
musk1	0.6213	0.7548	0.6874	0.6233	0.5578	0.4773	0.5008	0.6411	0.5091	0.6213
musk2	0.5626	0.5626	0.6596	0.6846	0.5409	**0.7378**	0.6043	0.5949	0.5572	0.5626
component	0.0000	0.0000	0.6587	0.5984	0.7608	0.5749	0.7558	0.0000	0.6773	0.0000
function	0.0000	0.0000	0.0000	0.5769	0.8164	0.0000	0.8061	0.0000	0.7130	0.0000
webmining	0.0000	0.0000	0.0000	0.2815	0.2020	0.0000	0.3327	0.0000	0.4915	0.0000
Average	0.3336	0.4870	0.5687	0.5824	0.6033	0.5464	0.5628	0.5082	0.5998	0.3389

Precision	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.5914	0.6074	0.9091	0.6836	0.6859	0.6767	0.5893	0.6089	0.8930	**0.9550**
westEast	0.2365	0.3588	**1.0000**	0.3757	0.5763	0.4228	0.2978	0.4390	0.9479	0.9633
suramin	0.6364	0.6000	0.5325	0.6364	0.7190	0.7484	**0.7778**	0.7284	0.7413	**0.7778**
mut-atoms	0.7453	0.8120	0.8179	0.7396	**0.8332**	0.8278	0.7941	0.8144	0.8133	0.8010
mut-bonds	0.7638	0.8588	0.8288	0.8609	0.8650	**0.8819**	0.8385	0.8603	0.8423	0.8431
mut-chains	0.7678	**0.8906**	0.8177	0.8676	0.8604	0.8904	0.8352	0.8855	0.8582	0.8754
trx	0.1623	0.5456	0.1000	0.6000	0.7109	0.9100	**1.0000**	0.8052	0.1056	0.0315
tiger	0.6663	0.6935	0.6966	**0.7395**	0.7156	0.7393	0.7156	0.7388	0.7316	0.7321
fox	0.5948	0.5996	0.6037	0.5847	0.6160	0.6421	0.6355	0.6084	0.6563	**0.6644**
elephant	0.7220	0.8255	0.7685	0.8642	0.8432	0.8312	0.7899	**0.8809**	0.8136	0.8380
musk1	0.5029	**0.8235**	0.8200	0.7081	0.6378	0.5912	0.7520	0.7015	0.7828	0.8092
musk2	0.5234	0.5816	0.5547	0.6766	0.5990	0.6431	0.6180	0.6511	0.4563	0.4550
component	0.5820	0.8494	0.7449	0.4045	0.5672	0.8751	0.7541	0.6274	0.8626	**0.8894**
function	0.6540	0.8515	0.7821	**0.9385**	0.8090	0.8829	0.8217	0.8468	0.8954	0.8965
webmining	0.4363	0.5084	0.3602	0.0000	0.3363	0.4796	0.5370	0.1696	0.8289	**0.8775**
Average	0.5723	0.6937	0.6891	0.6453	0.6917	0.7362	0.7171	0.6911	0.7486	**0.7606**

Table 4 shows the AUC trade-off between TPR and FPR, showing a similar behavior than reported for Kappa. It is noted how Bagging shows also good performance, achieving the best results for 4 out of 15 datasets, although MI-TLC obtains better average results. MVMI improves the MI-TLC results in all datasets except *mut-atoms*. On the other hand, CitationKNN obtains the worst results, yet it is still a classic method in multi-instance.

Tables 5 and 6 show precision and recall results, respectively. Precision and recall are two measures that when used together they provide a balance of the predictions. Results indicate that again, ensemble methods, MI-TLC, and MVMI obtain high precision, i.e. keeping low the number of false positives. MITI is the algorithm obtaining the best recall results, i.e having lower number of false negatives, but at the cost of having much

Table 6: Recall results.

Recall	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	0.5600	0.4200	0.6200	0.5600	0.6600	0.9000	0.9400	0.7400	0.6200	0.6000
westEast	0.1877	0.1877	0.1383	0.3358	0.4864	0.4420	0.3481	0.4938	0.2296	0.6049
suramin	**1.0000**	**1.0000**	0.9714	0.1429	0.4571	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9714
mut-atoms	0.8560	0.9696	0.8816	0.8928	0.9376	0.7632	**1.0000**	0.8288	0.8624	0.0992
mut-bonds	0.8544	0.8640	0.9248	0.9344	0.8368	0.7952	**1.0000**	0.8400	0.8864	0.0016
mut-chains	0.8624	0.9104	0.8864	0.8272	0.8464	0.8064	**1.0000**	0.8832	0.8976	0.1296
trx	0.0960	0.0000	0.4800	0.3280	0.1760	0.4160	0.0000	0.0000	0.3600	0.0160
tiger	0.0000	0.6140	0.6760	0.7520	0.7380	0.5540	0.7800	0.7980	0.7480	0.5560
fox	0.0000	0.7060	0.6180	0.6980	0.6460	0.5320	0.1720	0.5880	0.7320	0.0000
elephant	0.0000	0.7680	0.8300	0.7880	0.8680	0.7160	0.7200	0.8620	**0.9300**	0.0000
musk1	0.8589	0.7498	0.8908	0.8966	0.8126	0.8116	0.7932	0.8280	0.8357	0.8589
musk2	0.6000	0.6000	0.5641	**0.8615**	0.5385	0.6205	0.5641	0.5949	0.5795	0.6000
component	0.0000	0.0000	0.2473	0.1385	0.3872	0.1201	0.0293	0.0000	0.6378	0.0000
function	0.0000	0.0000	0.0000	0.1192	0.3431	0.0000	0.0483	0.0000	0.7002	0.0000
webmining	0.0000	0.0000	0.0000	0.0198	0.9676	0.0000	**1.0000**	0.0000	0.7734	0.0000
Average	0.3917	0.5193	0.5819	0.5530	0.6468	0.5651	0.6263	0.5638	0.7195	0.2958

Recall	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.7800	0.5800	**1.0000**	0.7400	0.7200	0.7600	0.5200	0.5600	0.8222	0.8600
westEast	0.4074	0.4173	**0.8025**	0.3259	0.3383	0.2469	0.1901	0.3877	0.4859	0.4938
suramin	**1.0000**	0.8571	0.7143	**1.0000**	0.7429	0.8286	**1.0000**	0.9143	0.9531	**1.0000**
mut-atoms	0.9216	0.8640	0.8336	0.8816	0.8784	0.8832	0.8752	0.8768	0.9147	0.9008
mut-bonds	0.9456	0.8736	0.8672	0.8816	0.8800	0.8960	0.8720	0.8832	0.9015	0.9024
mut-chains	0.9552	0.9008	0.9104	0.9120	0.8576	0.8960	0.8912	0.9152	0.9019	0.9200
trx	**0.9520**	0.2880	0.0080	0.0240	0.3440	0.2000	0.0640	0.1580	0.0312	0.0512
tiger	0.7580	0.7140	0.7480	0.7940	0.7440	**0.8100**	0.7240	0.7820	0.7915	0.7920
fox	**0.7920**	0.6000	0.6380	0.6220	0.6300	0.6780	0.6940	0.6200	0.7273	0.7363
elephant	0.8980	0.8420	0.7880	0.8760	0.8480	0.8860	0.8420	0.8580	0.8745	0.9008
musk1	0.8213	0.8744	0.7894	0.9246	0.8473	0.7430	0.8193	0.8918	0.9094	**0.9401**
musk2	0.6667	0.6564	0.6667	0.6103	0.6462	0.6923	0.6051	0.6308	0.3369	0.3360
component	**0.7660**	0.6374	0.6610	0.1792	0.3180	0.6227	0.5248	0.3653	0.6085	0.6274
function	0.8027	0.7029	0.7088	0.6546	0.6840	0.6896	0.5576	0.6564	0.8123	**0.8133**
webmining	0.8345	0.3622	0.4424	0.0000	0.1367	0.3165	0.3651	0.0342	0.4502	0.4766
Average	**0.8201**	0.6780	0.7052	0.6284	0.6410	0.6766	0.6363	0.6356	0.7014	0.7167

lower precision. Therefore, its behavior its not balanced and it is not a good performance trade-off, while MVMI achieves the best average precision and the second best average recall. Nevertheless, the performance of ensembles, MI-TLC, and MVMI remain competitive as compared with the other traditional learners. The next section extends the algorithms comparison to detect whether there are statistical significant differences among the performance of the classifiers.

4.3. Statistical analysis

Results are analyzed using multiple and pairwise non-parametric statistical tests [18]. Table 7 shows the ranking of the algorithms for each performance measure. The ranking was built following the philosophy of the

Table 7: Rank results.

Ranks	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
Accuracy	15.73	12.77	11.30	13.43	14.93	13.90	15.03	10.83	12.73	15.20
Kappa	15.90	14.07	11.93	13.43	13.97	13.90	14.77	11.57	11.30	15.70
AUC	16.80	11.97	11.60	13.60	12.07	15.00	16.23	13.57	13.47	15.50
Precision	16.00	13.57	12.03	13.67	13.00	13.67	13.77	11.57	12.50	14.93
Recall	15.43	13.90	12.30	10.57	11.13	13.53	9.80	12.70	8.30	15.60
Average	15.97	13.25	11.83	12.94	13.02	14.00	13.92	12.05	11.66	15.39

Ranks	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
Accuracy	13.87	8.27	9.30	7.70	7.10	5.73	7.47	6.63	4.87	**3.20**
Kappa	11.93	8.07	8.70	8.43	6.80	4.87	8.57	6.40	5.60	**4.10**
AUC	14.13	7.40	10.80	6.70	6.53	3.83	7.47	7.20	3.87	**2.27**
Precision	13.83	7.87	9.20	7.90	6.83	**4.87**	7.40	6.27	6.13	5.00
Recall	**4.33**	10.10	9.60	9.23	10.73	8.67	11.67	9.80	7.27	5.33
Average	11.62	8.34	9.52	7.99	7.60	5.59	8.51	7.26	5.55	**3.98**

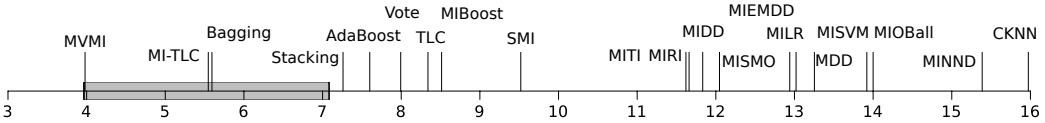

Figure 2: Bonferroni–Dunn test.

Friedman test [73] where the algorithm with the best value metric in one dataset is given a rank of 1 for that dataset, the algorithm with the next best metric value has the rank of 2, and so on. Finally, the average ranks of each algorithm in all datasets are calculated, which are shown in the table.

These ranks let us know which algorithm obtains the best results across all datasets. This way, the algorithm with the lowest and closest value to 1 indicates the best performance across the datasets and it is bold-faced. Specifically, MVMI obtains the best rank for accuracy, Kappa, and AUC, while Bagging does for precision and MITI does for recall. The average rank evaluates the overall performance of algorithms considering all the measures. Thus, MVMI is ranked first and it is considered the best performing, followed by MI-TLC, and the ensemble methods.

The Bonferroni–Dunn test is used to find the significant differences occurring between algorithms in multiple comparison. It assumes that the performance of two classifiers is significantly different if the corresponding average ranks differ by at least a critical difference value that is calculated based on the number of test cases. Given the number of datasets and algorithms compared, the Bonferroni-Dunn critical value is 3.08 for $\alpha = 0.05$. Therefore, algorithms with a ranking higher than 7.06 (MVMI rank + critical value) are said to perform statistically worse than MVMI. Statistical

Table 8: Wilcoxon, Nemenyi, and Holm tests.

MVMI *vs.*	Wilcoxon R^+	Wilcoxon R^-	Wilcoxon p-value	Nemenyi p-value	Holm p-value
CKNN	2689	86	2.4E-12	2.0E-16	0.00263
MDD	2733	116	7.6E-12	1.5E-13	0.00333
MIDD	2520	330	7.4E-09	3.3E-13	0.00454
MIEMDD	2523	327	6.8E-09	2.1E-13	0.00384
MILR	2574	276	1.3E-09	2.0E-13	0.00357
MIOBall	2468	307	5.9E-09	1.8E-13	0.00294
MISVM	2631	218	2.6E-10	2.1E-13	0.00312
MISMO	2650	124	1.6E-11	2.2E-13	0.00416
MIRI	2466	309	6.3E-09	6.3E-13	0.00500
MINND	2709	66	1.1E-12	2.0E-16	0.00277
MITI	2431	343	1.8E-08	7.8E-13	0.00555
TLC	2381	469	4.5E-07	0.00110	0.00833
SMI	2245	530	3.9E-06	1.8E-06	0.00625
Vote	2292	483	1.1E-06	0.00521	0.01000
AdaBoost	2345	504	1.1E-06	0.02496	0.01250
Bagging	1997	853	0.00255	0.98302	0.02500
MIBoost	2535	315	1.1E-08	0.00048	0.00714
Stacking	2208	567	9.9E-06	0.07984	0.01666
MI-TLC	2615	235	3.3E-10	0.98774	0.05000

differences are found for every algorithm except for Bagging and MI-TLC. Figure 2 shows the Bonferroni–Dunn test with the ranks of the algorithms. The critical distance is represented as a gray area, and those algorithms that exceed this area are said to perform statistically worse than MVMI.

The Wilcoxon, Nemenyi, and Holm tests were run to compare the pairwise performance between MVMI and the other algorithms. These non-parametric tests allow us to address whether there are significant differences between pairs of algorithms. This way, the null hypothesis maintains that there are no significant differences among the performance values, while the alternative hypothesis assures that there are. Table 8 shows the sum of ranks R^+ and R^- for the Wilcoxon rank-sum, and the p-values for the three tests, which indicate the statistical confidence instead of using a fixed α value.

Different tests provide complementary analysis of the results. According to the Wilcoxon test, MVMI is found to obtain statistically better performance than all the algorithms with p-values < 0.01. As for the Nemenyi test, statistical significant differences are identified for all traditional methods, while only some for ensembles approaches. Specifically, there are no differences found when comparing MVMI with Bagging and MI-TLC. According to the Holm's test, which is considered more powerful [18], there are significant differences and the p-values for these methods are ≤ 0.05. Consequently, the tests support the better performance of MVMI, and acknowl-

Table 9: Run time (seconds).

Run time	CKNN	MDD	MIDD	MIEMDD	MILR	MIOBall	MISVM	MISMO	MIRI	MINND
eastWest	0.69	4.04	2.29	0.60	**0.26**	0.30	0.40	0.37	0.31	0.88
westEast	0.01	1.55	0.98	0.12	0.09	0.01	0.10	0.03	0.02	0.12
suramin	1.22	19.59	1.26	1.44	0.07	0.25	2.01	0.25	0.31	0.18
mut-atoms	0.28	2.04	7.17	0.23	0.08	0.64	0.13	0.13	0.18	1572.52
mut-bonds	2.62	19.54	44.13	1.42	0.12	1.79	7.38	0.91	0.46	103.15
mut-chains	6.66	60.48	183.97	3.89	0.90	2.88	19.08	9.94	1.49	13.35
trx	94.36	910.51	1021.71	12.37	0.90	8.01	421.17	39.31	7.49	467.29
tiger	3.77	121.84	85.24	84.64	**0.07**	0.66	1.57	0.93	0.41	72.58
fox	4.12	367.90	101.08	83.80	0.46	0.92	2.62	1.16	0.28	3142.56
elephant	4.51	122.29	233.57	71.58	5.08	0.97	2.81	1.14	0.14	85.11
musk1	0.50	16.94	17.29	9.75	0.70	0.13	0.69	0.48	0.08	2.24
musk2	95.15	3610.99	243.22	124.20	68.73	5.77	69.04	5.18	9.13	8.44
component	1319.84	638.15	274.85	364.85	6.47	1320.76	2509.08	1665.65	276.85	179.75
function	3118.30	655.57	1320.37	242.31	16.80	4267.16	242.50	3165.65	315.43	774.40
webmining	307.94	188.39	251.72	81.55	497.05	684.83	375.38	724.81	9.20	39.27
Average	330.67	449.32	252.59	72.18	39.85	419.67	243.60	374.40	41.45	430.79
Ranking	12.63	17.40	17.07	12.83	5.73	9.33	12.83	10.57	5.77	13.83
Run time	MITI	TLC	SMI	Vote	AdaBoost	Bagging	MIBoost	Stacking	MI-TLC	MVMI
eastWest	0.29	0.72	0.32	0.41	1.33	0.80	0.67	1.04	0.34	0.90
westEast	0.02	0.03	**0.00**	0.04	0.43	0.22	0.11	0.27	0.16	0.04
suramin	0.23	0.10	**0.01**	0.12	0.88	0.79	0.77	1.10	1.18	0.15
mut-atoms	0.02	0.12	**0.01**	0.15	0.59	0.79	0.10	0.52	0.41	0.09
mut-bonds	0.08	0.42	**0.01**	0.54	4.15	2.74	0.81	3.48	1.82	0.76
mut-chains	0.23	0.50	**0.01**	0.69	7.69	5.51	1.96	5.86	5.79	1.71
trx	2.50	5.58	**0.01**	6.63	155.90	68.12	1.84	76.98	35.44	8.17
tiger	0.15	0.58	0.09	0.55	13.14	4.66	2.61	9.95	0.95	0.28
fox	0.19	0.61	**0.13**	0.64	16.34	8.07	4.69	12.38	0.57	0.82
elephant	0.12	0.50	**0.09**	0.60	13.18	5.94	4.60	8.63	0.83	0.54
musk1	0.06	0.16	**0.02**	0.19	3.04	3.94	0.18	2.97	0.20	0.26
musk2	2.33	5.90	**0.04**	6.33	89.12	54.24	32.36	80.71	12.04	10.78
component	38.50	32.45	**0.99**	30.04	2141.47	347.42	739.29	345.41	529.83	745.86
function	145.00	50.92	**1.03**	61.63	4546.63	613.99	1859.04	431.93	3654.22	2528.41
webmining	**6.64**	299.40	10.85	384.98	4690.59	4798.72	1177.49	2821.74	714.33	688.16
Average	13.09	26.53	**0.91**	32.90	778.97	394.40	255.10	253.53	330.54	265.80
Ranks	3.03	5.50	1.47	6.43	16.60	13.87	10.47	14.33	11.27	9.03

edges the second and third best results for MI-TLC and Bagging. On the other hand, MINND, CitationKNN, and MDD show the worst performance as compared with MVMI. Although these algorithms are quite popular in the literature, the experimental study clearly assesses that recent state-of-the art learners easily overcome these traditional multi-instance learning methods.

4.4. Run time analysis

Table 9 collects the run time of the algorithms in seconds. The fastest technique for each dataset is highlighted in bold-face, and the average and rank results are provided for each method. Simple MI (SMI) is the fastest algorithm due to its simplicity (reduces a bag into a single mono-instance),

whereas MDD and MIDD are the slowest according to the ranks. MVMI shows an average run time similar to the other ensemble methods, but when analyzing the ranks it obtains a much better value. Specifically, it is interesting to conduct a global analysis on the trade-off between quality of results and run time of ensemble methods. MVMI obtained the best quality results followed by Bagging and MI-TLC, and also shows to run faster. Globally, it is one of the fastest ensemble methods, yet traditional multi-instance classifiers are much faster at the cost of significantly worse quality results as evaluated previously in the statistical analysis.

It is necessary to point of that the computational complexity of MVMI relies on the computational complexity of its base classifiers when run on the views. As indicated in Section 3.2.1, MIMV trains in parallel the SimpleMI [56], MISMO [57], and TLC [56] base learners to identify the best performing method for each view. These learners show very fast run time in the table when applied independently to the multi-instance data.

4.5. Missing view data analysis

One of the advantages of the multi-view multi-instance representation defined in Section 3.1 is that it allows the views to represent partial information for the bags. This reflects real-world scenarios where data coming from different sources is not available for a given view, yet the classifiers must be able to learn from the partial data views provided. On the contrary, in single-view representation missing data is represented as missing values for the features of the views not available. Many traditional learners not capable of learning from missing values preprocess the data replacing the missing information with statistics (mean, median, etc), which may bias the data and provide incorrect predictions.

Figure 3 shows the accuracy of the top five performing algorithms when varying the percentage of missing data in the views. Average results for the 15 datasets is illustrated. This way, we analyze the performance impact when increasing the number of missing values (in single-view) and when removing the bag view data (in multi-view). For the sake of conducting a fair comparison, both single-view and multi-view representations removed the same feature values for the instances in the bag. It is clearly noticed the performance deterioration when increasing the amount of missing data. All approaches are observed to be affected very similar, but especially MI-TLC loses significant accuracy when increasing to 50% of missing data. MVMI keeps having the best accuracy when compared to the other methods.

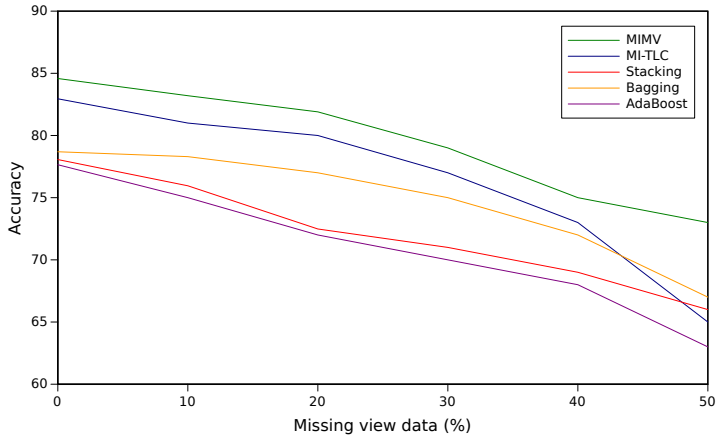

Figure 3: Analysis of accuracy deterioration when varying missing data in the views.

5. Conclusion

In this paper we proposed an ensemble approach to multi-view multi-instance learning that builds classification models on multiple heterogeneous data views. The ensemble was designed to take advantage of the complementary information distributed in multiple views represented using multi-instance bags, without conducting any multi-instance problem transformation. The best performing base classifiers are selected for the views, identifying the base learner with adapts best for the feature vectors in each view. The ensemble employed a voting scheme that weighted the predictions for each of the views based on the training error of the classifiers. This allowed to decrease the influence of data in certain views having low accurate classifiers or providing irrelevant/noisy information.

The experiments carried out support the better performance of the multi-view ensemble approach as compared with the base learners, ensemble methods, and a multi-view algorithm on a set of varied datasets and five different performance measures. Results were validated using a statistical analysis with non-parametric tests which indicate the statistically better performance of the multi-view ensemble approach and identifies worst performance methods for this problem. MVMI showed the better performance of learning on the original multi-view multi-instance data rather than the single meta-instance multi-view data transformation conducted by MI-TLC, while keeping competitive run times.

Acknowledgments

This research was supported by the Spanish Ministry of Economy and Competitiveness, project TIN2014-55252-P, and by FEDER funds.

References

[1] S. Sun, A survey of multi-view machine learning, Neural Computing and Applications 23 (2013) 2031–2038.

[2] H. Xue, S.-C. Chen, J. Liu, J.-J. Huang, Multi-view classification method based on cross-view constraints, Pattern Recognition and Artificial Intelligence 27 (2014) 97–102.

[3] S. Sun, J. Shawe-Taylor, L. Mao, PAC-Bayes analysis of multi-view learning, Information Fusion 35 (2017) 117–131.

[4] X. Zhu, X. Li, S. Zhang, Block-row sparse multiview multilabel learning for image classification, IEEE Transactions on Cybernetics 42 (2016) 450–461.

[5] F. Zou, Y. Liu, H. Wang, J. Song, J. Shao, K. Zhou, S. Zheng, Multi-view multi-label learning for image annotation, Multimedia Tools and Applications 75 (2016) 12627–12644.

[6] Y. Luo, T. Liu, D. Tao, C. Xu, Multiview matrix completion for multilabel image classification, IEEE Transactions on Image Processing 24 (2015) 2355–2368.

[7] E. L. Gibaja, J. M. Moyano, S. Ventura, An ensemble-based approach for multi-view multi-label classification, Progress in Artificial Intelligence 5 (2016) 251–259.

[8] F. Wu, Y. Huang, Z. Yuan, Domain-specific sentiment classification via fusing sentiment knowledge from multiple sources, Information Fusion 35 (2017) 26–37.

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

[10] S. Chen, L. Jiang, An empirical study on multi-instance learning, Advances in Information Sciences and Service Sciences 4 (2012) 193–202.

[11] J. Amores, Multiple instance classification: Review, taxonomy and comparative study, Artificial Intelligence 201 (2013) 81–105.

[12] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez-Tarragó, S. Vluymans, Multiple Instance Learning: Foundations and Algorithms, Springer, 2016.

[13] J. Foulds, E. Frank, A review of multi-instance learning assumptions, Knowledge Engineering Review 25 (2010) 1–25.

[14] S. Sabato, N. Tishby, Multi-instance learning with any hypothesis class, Journal of Machine Learning Research 13 (2012) 2999–3039.

[15] X. Wang, X. Liu, S. Matwin, N. Japkowicz, H. Guo, A multi-view two-level classification method for generalized multi-instance problems, in: IEEE International Conference on Big Data, 2014, pp. 104–111.

[16] Y. Lin, X. Hu, X. Wu, Ensemble learning from multiple information sources via label propagation and consensus, Applied Intelligence 41 (2014) 30–41.

[17] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, L. I. Kuncheva, Diversity techniques improve the performance of the best imbalance learning ensembles, Information Sciences 325 (2015) 98–117.

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

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

[20] R. Langone, J. Suykens, Supervised aggregated feature learning for multiple instance classification, Information Sciences 375 (2017) 234–245.

[21] Q. Tan, H. Deng, P. Yang, Knowledge transfer across different domain data with multiple views, Neural Computing and Applications 25 (2014) 15–23.

[22] N. Chen, J. Zhu, F. Sun, E. Xing, Large-margin predictive latent subspace learning for multiview data analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (2012) 2365–2378.

[23] C. Xu, D. Tao, C. Xu, Large-margin multi-view information bottleneck, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (2014) 1559–1572.

[24] M. Mayo, E. Frank, Experiments with multi-view multi-instance learning for supervised image classification, in: International Conference Image and Vision Computing, 2011, pp. 363–369.

[25] A. Fakeri-Tabrizi, M.-R. Amini, C. Goutte, N. Usunier, Multiview self-learning, Neurocomputing 155 (2015) 117–127.

[26] M. Volpi, G. Matasci, M. Kanevski, D. Tuia, Semi-supervised multiview embedding for hyperspectral data classification, Neurocomputing 145 (2014) 427–437.

[27] I. Triguero, S. García, F. Herrera, Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study, Knowledge and Information Systems 42 (2015) 245–284.

[28] X. Zhu, H.-I. Suk, Y. Zhu, K.-H. Thung, G. Wu, D. Shen, Multi-view Classification for Identification of Alzheimer's Disease, in: 6th International Workshop on Machine Learning in Medical Imaging, 2015, pp. 255–262.

[29] B. Wu, E. Zhong, A. Horner, Q. Yang, Music emotion recognition by multi-label multi-layer multi-instance multi-view learning, in: ACM International Conference on Multimedia, 2014, pp. 117–126.

[30] D. Zhang, J. He, R. Lawrence, MI2LS: Multi-instance learning from multiple information sources, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013, pp. 149–157.

[31] J. Wu, Z. Hong, S. Pan, X. Zhu, Z. Cai, C. Zhang, Exploring features for complicated objects: Cross-view feature selection for multi-instance learning, in: ACM International Conference on Information and Knowledge Management, 2014, pp. 1699–1708.

[32] W. Li, L. Duan, I. W.-H. Tsang, D. Xu, Co-labeling: A new multi-view learning approach for ambiguous problems, in: IEEE International Conference on Data Mining, 2012, pp. 419–428.

[33] C.-T. Nguyen, X. Wang, J. Liu, Z.-H. Zhou, Labeling complicated objects: Multi-view multi-instance multi-label learning, in: AAAI Conference on Artificial Intelligence, volume 3, 2014, pp. 2013–2019.

[34] L. Rokach, Ensemble-based classifiers, Artificial Intelligence Review 33 (2010) 1–39.

[35] V. Cheplygina, D. Tax, M. Loog, Dissimilarity-based ensembles for multiple instance learning, IEEE Transactions on Neural Networks and Learning Systems 27 (2016) 1379–1391.

[36] M. Wozniak, M. Graa, E. Corchado, A survey of multiple classifier systems as hybrid systems, Information Fusion 16 (2014) 3–17.

[37] N. C. Oza, K. Tumer, Classifier ensembles: Select real-world applications, Information Fusion 9 (2008) 4–20.

[38] P. V. Radtke, E. Granger, R. Sabourin, D. O. Gorodnichy, Skew-sensitive boolean combination for adaptive ensembles - an application to face recognition in video surveillance, Information Fusion 20 (2014) 31–48.

[39] Z. Zhou, M. Zhang, Solving multi-instance problems with classifier ensemble based on constructive clustering, Knowledge and Information Systems 11 (2007) 155–170.

[40] X. Kang, D. Li, S. Wang, A multi-instance ensemble learning model based on concept lattice, Knowledge-Based Systems 24 (2011) 1203–1213.

[41] V. G. Kaburlasos, T. Pachidis, A lattice-computing ensemble for reasoning based on formal fusion of disparate data types, and an industrial dispensing application, Information Fusion 16 (2014) 68–83.

[42] X.-S. Xu, X. Xue, Z.-H. Zhou, Ensemble multi-instance multi-label learning approach for video annotation task, in: ACM International Conference on Multimedia, 2011, pp. 1153–1156.

[43] F. Sener, N. Ikizler-Cinbis, Ensemble of multiple instance classifiers for image re-ranking, Image and Vision Computing 32 (2014) 348–362.

[44] L. Zhang, Q. Zhang, L. Zhang, D. Tao, X. Huang, B. Du, Ensemble manifold regularized sparse low-rank approximation for multiview feature embedding, Pattern Recognition 48 (2015) 3102–3112.

[45] C. Hong, J. Yu, J. You, X. Chen, D. Tao, Multi-view ensemble manifold regularization for 3D object recognition, Information Sciences 320 (2015) 395–405.

[46] X. Xie, S. Sun, Multi-view laplacian twin support vector machines, Applied Intelligence 41 (2014) 1059–1068.

[47] S. Karakatic, V. Podgorelec, Improved classification with allocation method and multiple classifiers, Information Fusion 31 (2016) 26–42.

[48] S. Tulyakov, S. Jaeger, V. Govindaraju, D. Doermann, Review of Classifier Combination Methods, Springer, 2008, pp. 361–386.

[49] L. Kuncheva, J. Rodríguez, A weighted voting framework for classifiers ensembles, Knowledge and Information Systems 38 (2014) 259–275.

[50] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (2012) 463–484.

[51] M. Galar, A. Fernández, E. Barrenechea, F. Herrera, DRCW-OVO: Distance-based relative competence weighting combination for One-vs-One strategy in multi-class problems, Pattern Recognition 48 (2015) 28–42.

[52] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, NMC: nearest matrix classification - A new combination model for pruning One-vs-One ensembles by transforming the aggregation problem, Information Fusion 36 (2017) 26–51.

[53] Y. Zhang, B. Liu, J. Cai, S. Zhang, Ensemble weighted extreme learning machine for imbalanced data classification based on differential evolution, Neural Computing and Applications (2016) 1–9.

[54] I. Visentini, L. Snidaro, G. L. Foresti, Diversity-aware classifier ensemble selection via f-score, Information Fusion 28 (2016) 24–43.

[55] J. Rodríguez, L. Kuncheva, C. Alonso, Rotation forest: A new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 1619–1630.

[56] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemannr, I. H. Witten, The WEKA Data Mining Software: An Update, SIGKDD Explorations 11 (2009) 10–18.

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

[58] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework, Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17 (2011) 255–287.

[59] J. Wang, et Jean-Daniel Zucker, J. daniel Zucker, Solving the multiple-instance problem: A lazy learning approach, in: International Conference on Machine Learning, 2000, pp. 1119–1125.

[60] O. Maron, T. Lozano-Pérez, A framework for multiple-instance learning, in: Conference on Advances in Neural Information Processing Systems, 1998, pp. 570–576.

[61] Q. Zhang, S. A. Goldman, Em-dd: An improved multiple-instance learning technique, in: Advances in Neural Information Processing Systems, 2001, pp. 1073–1080.

[62] P. Auer, R. Ortner, A boosting approach to multiple instance learning, in: European Conference on Machine Learning, 2004, pp. 63–74.

[63] X. Xu, A nearest distribution approach to multiple-instance learning, in: University of Waikato, Hamilton, NZ, 2001.

[64] L. Bjerring, E. Frank, Beyond trees: Adopting miti to learn rules and ensemble classifiers for multi-instance data, in: Australasian Joint Conference on Artificial Intelligence, 2011, pp. 41–50.

[65] J. Kittler, M. Hatef, R. P. W. Duin, J. Matas, On combining classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1998) 226–239.

[66] Y. Freund, R. E. Schapire, Experiments with a new boosting algorithm, in: International Conference on Machine Learning, 1996, pp. 148–156.

[67] D. H. Wolpert, Stacked generalization, Neural Networks 5 (1992) 241–259.

[68] J. Derrac, S. García, S. Hui, P. Suganthan, F. Herrera, Analyzing convergence performance of evolutionary algorithms: A statistical approach, Information Sciences 289 (2014) 41–58.

[69] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Information Sciences 250 (2013) 113–141.

[70] Z. Zhang, B. Krawczyk, S. García, A. Rosales-Pérez, F. Herrera, Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data, Knowledge-Based Systems 106 (2016) 251–263.

[71] A. Ben-David, About the relationship between ROC curves and Cohen's kappa, Engineering Applications of Artificial Intelligence 21 (2008) 874–882.

[72] J. Huang, C. X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Transactions on Knowledge and Data Engineering 17 (2005) 299–310.

[73] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, Journal of Machine Learning Research 7 (2006) 1–30.