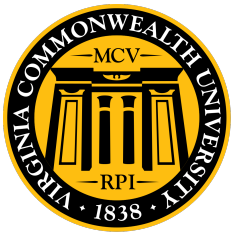
University of Córdoba, Spain
Virginia Commonwealth University, USA



Multi-label classification models for heterogeneous data: an ensemble-based approach

Modelos de clasificación multi-etiqueta para datos heterogéneos: un enfoque basado en ensembles

by

Jose María Moyano Murillo
Ph.D. Candidate

Advisors:

Dr. Eva L. Gibaja
Dr. Sebastián Ventura
Department of Computer Science and Numerical Analysis
University of Córdoba

Dr. Krzysztof J. Cios
Department of Computer Science
Virginia Commonwealth University

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy with a concentration in Computer Science at University of Córdoba (*Computación Avanzada, Energía y Plasmas* program) and Virginia Commonwealth University.

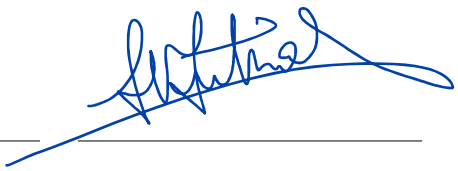
May 2020

The document entitled “Multi-label classification models for heterogeneous data: an ensemble-based approach”, reported by Jose María Moyano Murillo to qualify for the doctoral degree, has been conducted under the “Computación Avanzada, Energía y Plasmas” program at University of Córdoba and the program “Engineering, Doctor of Philosophy (Ph.D.) with a concentration in computer science/Computer Science Ph.D. with the University of Cordoba” at Virginia Commonwealth Univeristy, under the supervision of the doctors Eva Lucrecia Gibaja Galindo (University of Córdoba), Krzysztof J. Cios (Virginia Commonwealth University), and Sebastián Ventura Soto (University of Córdoba) fulfilling, in their opinion, the requirements demanded by this type of works and respecting the rights of other authors to be cited, when their results or publications have been used.

Córdoba, May 2020

Ph.D. Candidate

Jose María Moyano Murillo

Advisor	Advisor	Advisor
		
Dr. Eva L. Gibaja	Dr. Krzysztof J. Cios	Dr. Sebastián Ventura

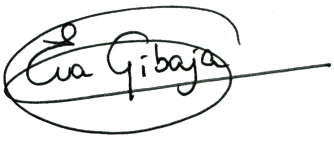
El documento llamado “Modelos de clasificación multi-etiqueta para datos heterogéneos: un enfoque basado en ensembles”, presentado por Jose María Moyano Murillo para optar al grado de doctor, ha sido realizado dentro del programa dual de doctorado en “Computación Avanzada, Energía y Plasmas” en la Universidad de Córdoba, y el programa “Engineering, Doctor of Philosophy (Ph.D.) with a concentration in computer science/Computer Science Ph.D. with the University of Cordoba” de la Virginia Commonwealth University, bajo la dirección de los doctores Eva Lucrecia Gibaja Galindo (Universidad de Córdoba), Krzysztof J. Cios (Virginia Commonwealth University), y Sebastián Ventura Soto (Universidad de Córdoba) cumpliendo, en su opinión, los requisitos exigidos a este tipo de trabajos y respetando los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Córdoba, Mayo de 2020

Doctorando

Jose María Moyano Murillo

Director



Dr. Eva L. Gibaja

Director

Dr. Krzysztof J. Cios

Director

Dr. Sebastián Ventura

TÍTULO DE LA TESIS:
Modelos de clasificación multi-etiqueta para datos heterogéneos: un enfoque basado en ensembles.

DOCTORANDO/A:
Jose María Moyano Murillo

INFORME RAZONADO DEL/DE LOS DIRECTOR/ES DE LA TESIS

(se hará mención a la evolución y desarrollo de la tesis, así como a trabajos y publicaciones derivados de la misma).

El trabajo realizado por el doctorando durante todo el periodo de investigación ha sido muy satisfactorio, como bien justifican los resultados obtenidos. Se comenzó trabajando en un artículo de revisión para mostrar una taxonomía de los distintos modelos de *ensembles* para clasificación multi-etiqueta, que permitiera una mejor comprensión del estado del arte, así como establecer reglas que permitan la mejor elección posible de modelos en determinadas circunstancias. Esta revisión fue publicada en la revista *Information Fusion*. Posteriormente, se comenzó a trabajar en el desarrollo de un primer modelo propio, basado en un algoritmo evolutivo, que codificaba un *ensemble* completo. El modelo produjo muy buenos resultados, siendo competitivo con los algoritmos que integran el estado del arte. Dicho modelo dio lugar a una segunda publicación en la revista *Information Fusion*. Con posterioridad a este modelo, se desarrolló un segundo modelo evolutivo con un enfoque diferente. En este caso, cada individuo codifica una parte del *ensemble*. Tras el proceso de evolución, se obtiene un ensemble combinado los individuos que mejor se comportan. Este segundo modelo se ha publicado en la revista *Knowledge-Based Systems*.

Es de destacar también que el trabajo de investigación desarrollado ha sido aceptado para su presentación en varios congresos científicos de prestigio internacional, tales como la *European Conference on Artificial Intelligence* o el *IEEE Congress on Evolutionary Computation*. Por último, indicar que esta tesis doctoral se realiza en cotutela con el Dpto. de Ciencias de la Computación e Inteligencia Artificial de la Virginia Commonwealth University, habiendo realizado el candidato una estancia de seis meses en dicha Universidad, bajo la supervisión del Dr. Krzysztof Cios, codirector de esta tesis.

Por todo lo comentado, consideramos que el trabajo presentado por D. José María Moyano puede calificarse de sobresaliente y que reúne con creces los requisitos exigibles a este tipo de trabajo.

Por todo ello, se autoriza la presentación de la tesis doctoral.

Córdoba, 27 de Mayo de 2020

Firma del/de los director/es

Fdo.: Krzysztof Cios Fdo.: Eva L. Gibaja Fdo.: Sebastián Ventura

Tesis con mención internacional

Esta tesis cumple con los criterios establecidos por la Universidad de Córdoba para la obtención del Título de Doctor con Mención Internacional.

1. Estancia predoctoral mínima de 3 meses fuera de España en una institución de enseñanza superior o centro de investigación de prestigio, cursando estudios o realizando trabajos de investigación relacionados con la tesis doctoral:

   Department of Computer Science, School of Engeneering, Virginia Commonwealth University, Richmond, Virginia, United States. Responsable de la estancia: **Dr. Krzysztof J. Cios**, Professor and Chair.

2. La tesis cuenta con el informe previo de dos doctores o doctoras expertos y con experiencia investigadora acreditada perteneciente a alguna institución de educación superior o instituto de investigación distinto de España:

   (a) **Dr. Isaac Triguero**. Associate Professor of Data Science, School of Computer Science, University of Nottingham, United Kingdom.

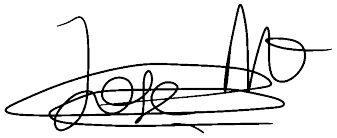
   (b) **Dr. João Gama**. Professor Catedratico, Faculdade de Economia, Universidade do Porto, Portugal.

3. Entre los miembros del tribunal evaluador de la tesis se encuentra un doctor procedente de una institución de educación superior distinto de España y diferente del responsable de la estancia predoctoral.

   Dr. Thang Dinh. Associate Professor, Department of Computer Science, Virginia Commonwealth University, Richmond, Virginia, United States.

4. Parte de la tesis doctoral se ha redactado y presentado en dos idiomas, castellano e inglés.

El doctorando



Jose María Moyano Murillo

This Doctoral Thesis has been partially funded by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund (FEDER), with projects **TIN2014-55252-P** and **TIN2017-83445-P**, as well as by the Spanish Ministry of Education under FPU Grant **FPU15/02958**. The stay at Virginia Commonwealth University has been also funded by the Spanish Ministry of Education, grant reference **EST18/00807**, and University of Córdoba (resolution published in **BOUCO no. 2019/00213**).

Acknowledgements

Writing these last words of the thesis, I am realizing how I have changed thorough the years since I started working in the KDIS group, back in 2012. Furthermore, which seemed to be a long period when I started the Ph.D. in 2016 comes to an end, and I am very pleased to have reached this objective. All the people mentioned below have contributed to make me who I am today.

First, I would like to thank every single one of my advisors. Sebastián and Eva gave me the opportunity to start a research career many years ago, even when I did not know what it entailed. Thanks for discovering me a field that I really enjoy. Krzysztof, thanks for the stay at your lab, during which I have learned a lot.

To the rest of people I have met in Richmond: Venant, Paolo, Javi, Chathu, and Dani. You made these months go faster and far more enjoyable. It will always be in my mind as good memories.

To all the people in the lab, who have not been few during all these years. Probably, trying to name all of you I would miss someone, but you make work days funnier, with an excellent working atmosphere. You guys are definitely friends beyond the lab.

To my family, who have been always supporting me and being proud of what I achieve. Thank you very much.

And last but not least, to Lucia. You not only accompany me everyday, but you also support me in everything I do and, far more important, motivate me to keep fighting for my goals and working hard. Thanks for your smiles and for every little dumb thing. And thank you for bringing Venus into our lives. I love you.

"Todo acto de bondad es una demostración de poderío".
Miguel de Unamuno.

Abstract

In recent years, the multi-label classification gained attention of the scientific community given its ability to solve real-world problems where each instance of the dataset may be associated with several class labels simultaneously. For example, in medical problems each patient may have several diseases at the same time, and in multimedia categorization, each item might be related with different tags or topics. Thus, given the nature of these problems, dealing with them as traditional classification where just one class label is assigned to each instance, would lead to a loss of information. The fact of having more than one label associated with each instance leads to new classification challenges that should be addressed, such as modeling compound dependencies among the labels, imbalance of the label space, and high dimensionality of the output space.

A large number of methods for multi-label classification was proposed in the literature, including several ensemble-based methods. Ensemble learning is a technique which is based on combining the outputs of many diverse base models, in order to outperform each separate model. In multi-label classification, ensemble methods are those that combine the predictions of several multi-label classifiers, and they were shown to outperform simpler multi-label classifiers. Therefore, given its performance, we focus our research on the study of ensemble-based methods for multi-label classification.

The first objective of this dissertation is to perform a thorough review of the state-of-the-art ensembles of multi-label classifiers. Its aim is twofold: 1) study different ensembles of multi-label classifiers proposed in the literature, and categorize them according to their characteristics, and proposing a novel taxonomy; and 2) perform an experimental study to find the method or family of methods, that perform better depending on the characteristics of the data, as well as provide guidelines for selecting the best method according to the characteristics of a given problem.

Since most of the ensemble methods for multi-label classification are based on creating diverse members by randomly selecting instances, input features, or la-

bels, our main objective is to propose novel ensemble methods for multi-label classification where the characteristics of the data are taken into account. For this purpose, we first propose an evolutionary algorithm able to build an ensemble of multi-label classifiers, where each of the individuals of the population is the entire ensemble. This approach is able to model the relationships among the labels with a relative low complexity and imbalance of the output space, while also using considering these characteristics to guide the learning process. Furthermore, it looks for an optimal structure of the ensemble considering its predictive performance and the number of times that each label appears in it. In this way, all labels are expected to appear a similar number of times in the ensemble, not neglecting any of them regardless of their frequency.

Then, we develop a second evolutionary algorithm to build ensembles of multi-label classifiers, but in this case each individual of the population is a hypothetical member of the ensemble, and not the entire ensemble. The fact of evolving members of the ensemble separately makes the algorithm less computationally complex and able to determine the quality of each member separately. However, a method to select the ensemble members needs to be defined. The process selects those classifiers that are both accurate and diverse to form an ensemble; we also control that all labels appear a similar number of times in the final ensemble.

In all experiments, the methods are compared using rigorous experimental setups and statistical tests over many evaluation metrics and reference datasets for multi-label classification. The experiments show that the proposed methods obtain significantly better and more consistent performance than the state-of-the-art methods of multi-label classification.

Resumen

En los últimos años, el paradigma de clasificación multi-etiqueta ha ganado atención en la comunidad científica, dada su habilidad para resolver problemas reales donde cada instancia del conjunto de datos puede estar asociada con varias etiquetas de clase simultáneamente. Por ejemplo, en problemas médicos cada paciente puede estar afectado por varias enfermedades a la vez, o en problemas de categorización multimedia, cada ítem podría estar relacionado con varias etiquetas o temas. Dada la naturaleza de estos problemas, tratarlos como problemas de clasificación tradicional donde cada instancia puede tener asociada únicamente una etiqueta de clase, conllevaría una pérdida de información. Sin embargo, el hecho de tener más de una etiqueta asociada con cada instancia conlleva la aparición de nuevos retos que deben ser abordados, como modelar las dependencias entre etiquetas, el desbalanceo de etiquetas, y la alta dimensionalidad del espacio de salida.

En la literatura se han propuesto un gran número de métodos para clasificación multi-etiqueta, incluyendo varios basados en *ensembles*. El aprendizaje basado en *ensembles* combina las salidas de varios modelos más simples y diversos entre sí, de cara a conseguir un mejor rendimiento que cada miembro por separado. En clasificación multi-etiqueta, se consideran *ensembles* aquellos métodos que combinan las predicciones de varios clasificadores multi-etiqueta, y estos métodos han mostrado conseguir un mejor rendimiento que los clasificadores multi-etiqueta sencillos. Por tanto, dado su buen rendimiento, centramos nuestra investigación en el estudio de métodos basados en *ensembles* para clasificación multi-etiqueta.

El primer objetivo de esta tesis el realizar una revisión a fondo del estado del arte en *ensembles* de clasificadores multi-etiqueta. El objetivo de este estudio es doble: I) estudiar diferentes *ensembles* de clasificadores multi-etiqueta propuestos en la literatura, y categorizarlos de acuerdo a sus características proponiendo una nueva taxonomía; y II) realizar un estudio experimental para encontrar el método o familia de métodos que obtiene mejores resultados dependiendo de las características de los datos, así como ofrecer posteriormente algunas guías para seleccionar el mejor método de acuerdo a las características de un problema dado.

Dado que la mayoría de *ensembles* para clasificación multi-etiqueta están basados en la creación de miembros diversos seleccionando aleatoriamente instancias, atributos, o etiquetas; nuestro segundo y principal objetivo es proponer nuevos modelos de *ensemble* para clasificación multi-etiqueta donde se tengan en cuenta las características de los datos. Para ello, primero proponemos un algoritmo evolutivo capaz de generar un *ensemble* de clasificadores multi-etiqueta, donde cada uno de los individuos de la población es un *ensemble* completo. Este enfoque es capaz de modelar las relaciones entre etiquetas con una complejidad y desbalanceo de etiquetas relativamente bajos, considerando también estas características para guiar el proceso de aprendizaje. Además, busca una estructura óptima para el *ensemble*, no solo considerando su capacidad predictiva, pero también teniendo en cuenta el número de veces que aparece cada etiqueta en él. De este modo, se espera que todas las etiquetas aparezcan un número de veces similar en el *ensemble*, sin despreciar ninguna de ellas independientemente de su frecuencia.

Posteriormente, desarrollamos un segundo algoritmo evolutivo capaz de construir *ensembles* de clasificadores multi-etiqueta, pero donde cada individuo de la población es un hipotético miembro del *ensemble*, en lugar del *ensemble* completo. El hecho de evolucionar los miembros del *ensemble* por separado hace que el algoritmo sea menos complejo y capaz de determinar la calidad de cada miembro por separado. Sin embargo, también es necesario definir un método para seleccionar los miembros que formarán el *ensemble*. Este proceso selecciona aquellos clasificadores que sean tanto precisos como diversos entre ellos, también controlando que todas las etiquetas aparezcan un número similar de veces en el *ensemble* final.

En todos los estudios experimentales realizados, los métodos han sido comparados utilizando rigurosas configuraciones experimentales y test estadísticos, involucrando varias métricas de evaluación y conjuntos de datos de referencia en clasificación multi-etiqueta. Los experimentos confirman que los métodos propuestos obtienen un rendimiento significativamente mejor y más consistente que los métodos en el estado del arte. Además, se demuestra que el segundo algoritmo propuesto es más eficiente que el primero, dado el uso de individuos representando clasificadores por separado.

Preface

The Spanish legislation for Ph.D. studies, RD 99/2011, published on January 28th, 2011 (BOE-A-2011-2541), grants each Spanish University competencies to establish the necessary supervision and evaluation procedures to guarantee the quality of Ph.D. theses.

Accordingly, the University of Córdoba has a specific regulation for Ph.D. studies, approved on December 21st, 2011. This regulation establishes two different modalities to elaborate the dissertation that the student has to present at the end of the doctorate studies. This Ph.D. thesis follows the modality described in the article no. 24 of the aforementioned regulation, referred as Ph.D. thesis as a compendium of publications. According to that article, the Ph.D. thesis can be presented as a compendium of, at least, three research articles published (or accepted for publication) in research journals of high quality, i.e. appearing in the first three quartiles of the Journal Citation Reports (JCR). If such a requirement is fulfilled, the manuscript has to include: an introduction to justify the thematic cohesion of the Ph.D. Thesis; the hypotheses and objectives to be achieved, and how they are associated to the publications; full copy of the publications; and conclusions.

Following these guidelines, this Ph.D. thesis is organized as follows. First, Chapter 1 introduces the dissertation, including background and state-of-the-art work in topics related with the thesis. Next, the objectives and motivations are presented in Chapter 2, indicating which of the journal papers associated with the thesis fulfill each of the objectives. Later, Chapters 3 to 5 present the main contributions of the thesis. In order to enhance the cohesion of the document, each of these chapters is divided in two parts: first, a brief introduction of the objectives, methodology, and conclusions of the corresponding study are presented; and then, the journal paper is directly included. Chapter 6 discusses the conclusions obtained thorough the dissertation as well as presents some lines and future work. Finally, in Chapter 7, other publications associated with this Ph.D. thesis are included. At the end of the document, the Vita of the Ph.D. candidate is also provided.

Acronyms and abbreviations

Acronym	Meaning
avgIR	Average Imbalance Ratio
BP-MLL	Back-Propagation for Multi-Label Learning
BR	Binary Relevance
Card	Cardinality
CBMLC	Clustering-Based for Multi-Label Classification
CC	Classifier Chains
CCEA	Cooperative CoEvolutionary Algorithm
CDE	Chi-Dep Ensemble
D3C	Dynamic selection and Circulating Combination-based Clustering
Dim	Dimensionality
Div	Diversity
DNF	Did Not Finish
EA	Evolutionary Algorithm
EAGLET	Evolutionary AlGorithm for multi-Label Ensemble opTimization
EBR	Ensemble of Binary Relevance
ECC	Ensemble of Classifier Chains
ELP	Ensemble of Label Powersets
EME	Evolutionary Multi-label Ensemble
EMLC	Ensemble of Multi-Label Classifiers
EMLS	Ensemble of Multi-Label Sampling
EPS	Ensemble of Pruned Sets
ExAcc	Example-based Accuracy
ExF	Example-based FMeasure
ExP	Example-based Precision
ExR	Example-based Recall
ExS	Example-based Specificity

Acronym	Meaning
GACC	Genetic Algorithm for ordering Classifier Chains
G3P	Grammar-Guided Genetic Programming
GA	Genetic Algorithm
GEP	Gene Expression Programming
GP	Genetic Programming
GPL	GNU General Public License
HL	Hamming Loss
HOMER	Hierarchy Of Multi-label classifiERs
IBLR-ML	Instance-Based learning by Logistic Regression for Multi-Label classification
LIFT	Label specIfic FeaTures for multi-label learning
LP	Label Powerset
MaAcc	Macro-averaged Accuracy
MaF	Macro-averaged FMeasure
MaP	Macro-averaged Precision
MaR	Macro-averaged Recall
MaS	Macro-averaged Specificity
MiAcc	Micro-averaged Accuracy
MiF	Micro-averaged FMeasure
MiP	Micro-averaged Precision
MiR	Micro-averaged Recall
MiS	Micro-averaged Specificity
MLC	Multi-Label Classification
MLDA	Multi-Label Dataset Analyzer
ML-ELM-RBF	Multi-Layer Extreme Learning Machine of Radial Basis Functions
ML-*k*NN	Multi-Label k-Nearest Neighbors
MLL	Multi-Label Learning

Acronym	Meaning
MLS	Multi-Label Stacking
MLSOL	Multi-Label Synthetic Oversampling based on the Local distribution
PCT	Predictive Clustering Tree
PS	Pruned Sets
PSO	Particle Swarm Optimization
RA*k*EL	RAndom k-labELsets
rDep	Ratio of Dependent labels pairs
RF-PCT	Random Forest of Predictive Clustering Trees
SA	Subset Accuracy
TREMLC	Triple Random Ensemble for Multi-Label Classification

Notation

Symbol	Meaning
$\mathcal{X}$	Input space
$\mathcal{Y}$	Output (label) space
$\mathcal{D}$	Multi-label dataset
$\mathbf{x}_i$	Set of input attributes for *i*th instance
Y_i	Set of relevant labels for *i*th instance
$\overline{Y}_i$	Set of irrelevant labels for *i*th instance
$\hat{Y}_i$	Predicted set of relevant labels for *i*th instance
$\overline{\hat{Y}}_i$	Predicted set of irrelevant labels for *i*th instance
λ_l	*l*th label
f_l	Frequency of *l*th label in $\mathcal{D}$
m	Number of instances
d	Number of input attributes
q	Number of labels
$\mathcal{H}$	Search space of models
h	Predictive model
$\mathbf{e}$	Ensemble
e^j	*j*th member of the ensemble
n	Number of classifiers in the ensemble
$\hat{\mathbf{b}}$	Bipartition provided by a model
b_{jl}	Bipartition provided by *j*th member of the ensemble for *l*th label
t	Threshold for ensemble prediction
a_{min}	Minimum number of times that each label should appear in the ensemble
$\mathbf{eV}$	Expected number of votes or appearances for each label in the ensemble

Symbol	Meaning
p	Population of the EA
s	Offspring individuals in the EA
ind	Individual of the population in the EA
$popSize$	Size of the population in the EA
G	Maximum number of generations of the EA
n_T	Number of classifiers built in the EA
β	Multiplier for fitness combination
ϕ	Correlation coefficient between labels
k	Size of the *k*-labelsets (subsets of *k* labels)
α	Significance level for statistical tests
Δ	Operator that returns the symmetric difference between two binary sets
$[\![\pi]\!]$	Operator that returns 1 if predicate π is true, and 0 otherwise

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter we provide an introduction to the topics on which this Ph.D. thesis is based. First, we provide a description of the ensemble learning framework, including main approaches to build ensembles (Section 1.1). Then, the Multi-Label Classification (MLC) paradigm is comprehensively explained (Section 1.2), including a formal definition, state-of-the-art MLC methods, characterization metrics, benchmark datasets, evaluation metrics to assess the performance of the methods, and an introduction to different multi-label libraries and tools. Finally, Evolutionary Algorithms (EAs), which are later used in our methodology, are described (Section 1.3).

1.1 Ensemble learning

When making crucial decisions, the tendency of humans is to gather information and opinions from different sources, then combine them into a final decision which is hoped to be better and more consistent than considering just one opinion [1]. Based on this reasoning, ensemble learning is a machine learning technique which combines predictions of individual learners from heterogeneous or homogeneous modeling to obtain a combined learner that improves the overall generalization ability and reduces the overfitting of each [2, 3].

Ensemble methods are considered as state-of-the-art for solving a wide range of machine learning problems, such as classification [4], regression [5], and clustering [6]. They were successfully used in fields such as finance [7], bioinformatics [8], medicine [9], and image retrieval [10].

There are three main reasons why an ensemble learner use to perform better than single learners [11]. First, when picking only one single learner we run the risk of selecting one out of several learners that get same performance on training data, but which may perform different over unseen data; therefore, selecting some of these learners and combining their outputs, the optimal one, i.e., this which performs better on unseen data, would be easily reached. In Figure 1.1a, the search space $\mathcal{H}$ of models is indicated with the outer line, while the inner line denotes the set of learners that get the same performance in training, and h^* denotes the optimal learner. Second, many learning algorithms use local search to find a model and they may not find the optimal learner, so running several times the learning algorithm and combining the obtained models may result in a better approximation to the optimal learner than any single one (Figure 1.1b). Third, since in most machine learning problems the optimal function might not be found, the optimal learner may be approximated by combining several feasible learners (Figure 1.1c).

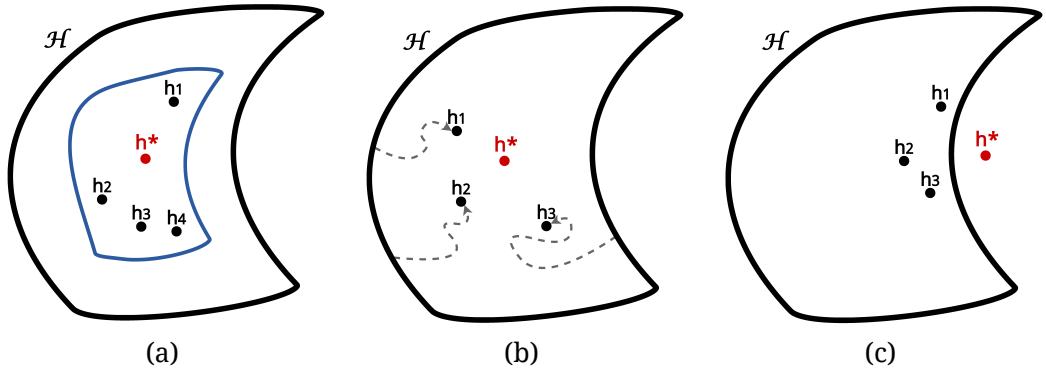


Figure 1.1: Reasons why ensemble methods outperform single learners. (a) Combining several learners that perform similarly on training data, the optimal one could be reached. (b) Combining several learners obtained by local search could better reach the optimal learner. (c) Combining several feasible learners, the non-reachable optimal one could be approximated.

1.1.1 Building phase

Although ensemble models tend to improve the generalization ability of base learners, those learners to be combined into an ensemble should be carefully chosen. Using accurate learners seems to be an obvious approach; however, an ensemble including learners that are very similar to each other could not perform as well

as expected, and it may perform even worse than individual ones. Therefore, diverse learners are usually preferred to be combined, although formal proof of this dependency does not exist [12, 13].

The main approaches that have been proposed to build ensembles of diverse learners could be categorized in the following groups [1]:

Input manipulation Each base model is built over a slightly different subset of the original training dataset (sampled with or without replacement). Thus, learners are diverse among them since each is focused on different input data.

Manipulated learning algorithm The way in which the learning algorithm performs is modified. Usually, different hyperparameters are used for each ensemble member, such as using neural networks with different learning rates or number of layers, or support vector machines with different regularization parameters or kernels.

Partitioning The original dataset is divided in several mutually exclusive subsets, and each of them is used to train a different base learner. The original data can be split following two different approaches: a) horizontal partitioning, where each subset is composed of different instances (being non-overlapping subsets, unlike in *input manipulation*); and b) vertical partitioning, where each member is built over all instances but each of them considering a different subset of the input attributes.

Output manipulation The output attribute (or attributes) is manipulated at each of the members of the ensemble. Depending on the problem, the output attributes would be categorical classes, real-valued targets, etc.

Hybridization Usually, instead of just using one approach, two or even more approaches are hybridized in order to obtain far more diverse learners, aiming to improve the overall predictive performance of the ensemble learner.

1.1.2 Prediction phase

Given an ensemble model **e** composed by n base learners, the final decision is made by combining individual predictions from each member e^j, being $j \in [1, n]$. Several approaches have been proposed in order to combine or integrate these predictions into the final one.

Weighting methods are the most usual techniques to combine predictions. These methods give a weight to each individual prediction, and then they are combined in any way. In particular, majority voting is the simplest and widely used weighting method, where the final prediction is the output with more votes among the base learners, or the average of predictions in real-valued outputs [5]. However, more complex approaches exist, such as giving a weight to each base learner based on their individual performance [14]; thus, the final prediction is biased by more accurate base learners while still considering all predictions.

On the other hand, instead of directly combining the predictions of individual learners, meta-learning methods build an ensemble in two-stages. In the first stage, several base learners are built and their predictions are gathered. Then, in a second phase, predictions of individual learners are used to build a meta-model, which may either extend the original input space with the predictions of previous stage, or just use these predictions as input attributes for the meta-model. Then, the output of the meta-model is used as the final ensemble prediction [15]. While weighting methods are better applicable in cases where the performance of base learners is similar, meta-learning methods are able to detect if certain base methods perform poorly in some subspaces.

1.2 Multi-label classification

Classification is one of the most popular and widely studied tasks in data mining. Traditional classification is a supervised learning task whose aim is to learn from data and their attributes to build a model that predicts the corresponding class for each of the instances. Whether binary (two classes) or multi-class (several classes), the main characteristic of traditional classification problems is that each of the in-

stances or examples is associated with one and only one class of the available ones. For example, as seen in Figure 1.2, each iris flower could belong to only one of their species, either *setosa*, *virginica*, or *versicolor*, so the model predicts one of these classes given the characteristics of the flower [16].

Figure 1.2: Example of traditional classification.

However, there exist a large number of classification tasks where each example may have not only one but several classes or labels associated with it simultaneously. For example, in multimedia annotation or text categorization problems, each item could be categorized using several labels or topics [17, 18]; in recommender systems, the user may receive more than one recommendation at a time [19]; in medical problems, patients may be affected by more than one disease [20]; and in biology, genes could be annotated with more than one function simultaneously [21]. MLC aims to build models to predict all the associated labels with each instance of the problem, and it has gained a lot of attention in the last decade [22].

In Figure 1.3, the image can be annotated with several from previously defined labels, such as *animal*, *raccoon*, and *wood*. As it can be observed, labels may be also related among them: if *raccoon* label is present, the *animal* label is very likely to be present too (although it does not need to be true in the other direction, i.e., the appearance of the *animal* label may not imply that the *raccoon* label is present), but it is very unlikely to the *snow* label to be relevant. Dealing with this image as a traditional classification problem (a.k.a. single-label classification), it would lead to a lose of information, since only one of the labels could be present.

Figure 1.3: Example of multi-label classification.

The MLC framework has been successfully applied to a large number of real-world problems, such as text categorization [23], multimedia annotation [24, 25, 26, 27], bioinformatics [8, 9, 28, 29], and social networks mining [30]. Therefore, the utility of the MLC paradigm in real-world problems have turned it into one of the hottest topics in data mining in the last years.

1.2.1 Formal definition of MLC

Let $\mathcal{D}$ be a multi-label dataset composed by a set of m instances, and defined as $\mathcal{D} = \{(\mathbf{x}_i, Y_i)|1 \leq i \leq m\}$. Let $\mathcal{X} = X_1 \times \cdots \times X_d$ be the d-dimensional input space, and $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ the output space composed by $q > 1$ labels. Each multi-label instance is composed by an input vector $\mathbf{x}_i$ and a set of relevant labels associated with it $Y_i \subseteq \mathcal{Y}$. Note that each different Y is also called labelset [22]. In Table 1.1, an example of multi-label dataset is shown. As seen, each example is labeled with one or more than one labels simultaneously.

Table 1.1: Example of multi-label dataset.

Instance	Features	Labels				
		λ_1	λ_2	λ_3	...	λ_q
#1	$\mathbf{x}_1$	0	1	0	...	0
#2	$\mathbf{x}_2$	1	0	1	...	1
#3	$\mathbf{x}_3$	0	1	0	...	1
#4	$\mathbf{x}_4$	0	1	0	...	0
#5	$\mathbf{x}_5$	0	1	1	...	0
...	...			...		
#m	$\mathbf{x}_m$	1	0	1	...	1

The goal of MLC is to construct a predictive model $h : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$ which provides a set of relevant labels for an unknown instance. Thus, for each $\mathbf{x} \in \mathcal{X}$, a bipartition $\hat{\mathbf{b}} = \left(\hat{Y}, \overline{\hat{Y}}\right)$ of the label space $\mathcal{Y}$ is provided, where $\hat{Y} = h(\mathbf{x})$ is the set of relevant labels and $\overline{\hat{Y}}$ the set of irrelevant ones. This bipartition could be also given as a binary vector $\hat{\mathbf{b}} = \{0, 1\}^q$, indicating if each label is relevant (1) or not (0).

Furthermore, lets define an Ensemble of Multi-Label Classifiers (EMLC) as a set of n multi-label classifiers. Each of the base classifiers h_j provides prediction $\hat{\mathbf{b}}_j = \{\hat{b_{j1}}, \hat{b_{j2}}, \ldots, \hat{b_{jq}}\}$ for all (or part of) the labels, each b_j being either 1 if the label is relevant and 0 otherwise. The final prediction of the ensemble is usually calculated given the average value of the predictions for each label $\hat{\mathbf{v}} = \{\hat{v_1}, \hat{v_2}, \ldots, \hat{v_q}\}$, where the $\hat{v_l}$ for each label λ_l is calculated as $\hat{v_l} = \frac{1}{n}\sum_{j=1}^{n} \hat{b_{jl}}$. However, many other methods instead of simple voting could be used in order to combine predictions in the ensemble [31].

1.2.2 Main challenges to address in multi-label data

Given the fact that each instance of the data may have more than one label associated, poses new challenges that need to be addressed, such as modeling the compound relationships among labels, the high dimensionality of the output space, and the imbalance of the output space.

Relationship among labels

As seen in Figure 1.3, output labels are not usually independent but they tend to be related to each other; e.g., a label may appear more frequently with some labels than with others. Thus, if the model is able to learn from these dependencies, its predictive performance would be improved. Consider that for a given instance, the learning method initially predicts just *raccoon* and *wood* labels as relevant ones. If the method models the relationship among labels, and it has learn that with a very high probability when the *raccoon* label appears, the *animal* label appears too, it would be able to amend its prediction and finally predict the *animal* label too. However, not learning these dependencies would lead to a poorer predictive performance.

Considering the way in which the MLC methods address this problem, they are usually categorized as: I) first-order strategies, where labels are modeled completely independent, and therefore the dependencies among labels are not learned; II) second-order strategies, where the dependencies among pairs of labels are taken into account; and III) high-order strategies, which model the dependencies among groups of more of two labels jointly [32].

Label imbalance

It is intrinsic to many problems that labels do not appear with the same frequency in the dataset. For example, in the dataset containing the image in Figure 1.3, maybe most of the instances have associated the *animal* label, but less have the *raccoon* one, and just a very small percentage of images are assigned the *snow* label. Thus, if the *snow* label barely appear in the dataset and the learning algorithm does not control this imbalance, it may be despised and the final method could not be able to predict it correctly.

The fact of having a lot of information for some labels but very little information of some others, leads to the difficulty of learning the infrequent ones. The task of learning from imbalanced datasets has been widely tackled in the literature; however, in multi-label scenarios, it should be addressed differently due to the high number of output labels.

Dimensionality of the output space

In multi-label scenarios, the dimensionality of the dataset is not only related to the number of instances and/or attributes as in many machine learning tasks, but also to the number of output labels. Problems with a low number of labels are easier to tackle, but in cases with a complex output space, the problem becomes far more difficult to be solved.

For example, it is much easier to learn the dependencies among labels if there are just six labels rather than a hundred labels. So, the dimensionality of the output space should be carefully considered in order to not to have extremely complex models.

1.2.3 Multi-label classification algorithms

In this section we introduce state-of-the-art methods in MLC, which are categorized into three main groups: problem transformation (PT), algorithm adaptation (AA), and EMLCs [22, 33]. Note that not all methods are able to deal with each of the problems or characteristics of multi-labeled data (such as imbalance, relationship among labels, and high dimensionality of the output space); for example, some are not able to learn from the dependencies among labels in any way to enhance the final prediction. Furthermore, several of those that are able to deal with any characteristic, still do not consider it in their building phase, e.g., using the relationships among labels to lead the learning process, towards combinations of related labels for example. In Table 1.2 a summary of the methods is provided, indicating for each of them if they are able to deal with (D) and/or consider these characteristics in the building phase (B).

Problem transformation methods

Problem transformation methods transform a multi-label problem into one or several single-label problems, then solving each new problem using traditional single-label algorithms. For ease of understanding, and considering the multi-label dataset in Table 1.1, schemes of main transformations are presented.

One of the most popular methods is Binary Relevance (BR) [34] which decomposes the multi-label learning problem into q independent binary classification problems, one for each label, as presented in Figure 1.4. The final multi-label prediction is obtained by combining the predictions of each single-label classifier. The fact that BR treats each label separately makes it simple, highly parallelizable, and resistant to overfitting label combinations, but it does not take into account label combinations so makes it unable to model possible dependencies among the labels. Thus, BR does not deal with any of the previously described problems in MLC.

Label specIfic FeaTures for multi-label learning (LIFT) [35] is based on the idea of modeling each label just considering their related input features, i.e., their label-specific features, thus avoiding the noise provided by features that are not cor-

Table 1.2: Summary of state-of-the-art MLC methods. It is indicated with a 'D' if the method is able to deal with the corresponding problem (imbalance, relationships among labels, and high dimensionality of the output space), and with a 'B' if it considers this characteristic at building phase.

		Imbalance	**Relationships**	**Output Dim.**
PTs	BR	-	-	-
	CC	-	D	-
	GACC	-	D, B	-
	LIFT	D, B	-	-
	LP	-	D	-
	PS	D, B	D	D, B
	ChiDep	D	D, B	D, B
AAs	PCT	-	D, B	-
	ML-*k*NN	-	D, B	-
	IBLR-ML	-	D, B	-
	BP-MLL	-	D, B	-
EMLCs	EBR	-	-	-
	ECC	-	D	-
	MLS	-	D	-
	HOMER	D	D, B	D, B
	AdaBoost.MH	-	-	-
	D3C	-	-	-
	EPS	D, B	D	D, B
	RA*k*EL	D	D	D, B
	TREMLC	D	D	D, B
	CDE	D	D, B	D, B
	RF-PCT	-	D, B	-
	CBMLC	D	D	D
	EMLS	D, B	D	-

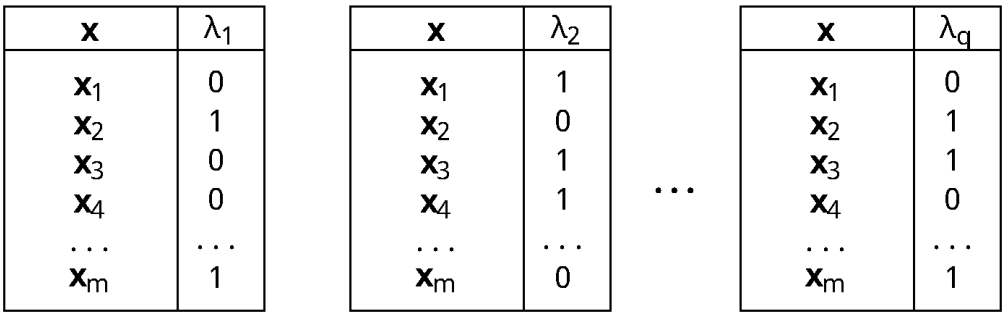


Figure 1.4: BR transformation.

related with them. For this purpose, LIFT uses clustering of the feature space to select a subset of features that best discriminate each label independently, and then builds a binary model for each of the labels. Figure 1.5 presents this transformation, where each binary classifier is also built over different subset of input features ($\mathbf{x}^{\lambda_l}$). In this process, LIFT considers the imbalance of the labels, being also able to deal with it; however, given the construction of independent models, it does not manage to deal neither with the relationship among labels nor the high dimensionality of the output space.

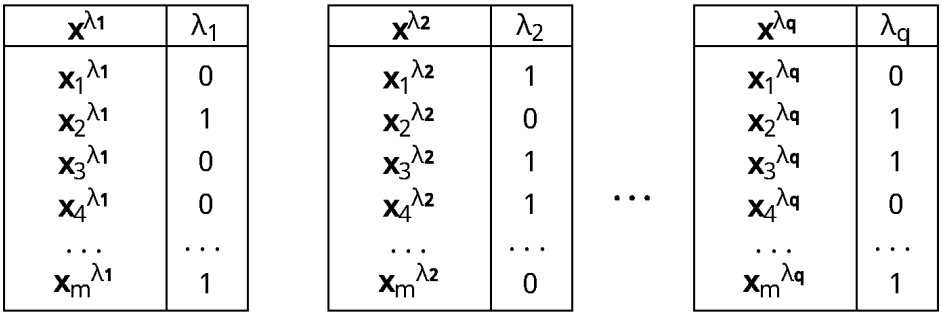


Figure 1.5: LIFT transformation.

In order to overcome the label independence assumption of BR, Classifier Chain (CC) [36] generates q binary classifiers but linked in such a way that each binary classifier also includes the label predictions of previous classifiers in the chain as additional input features (Figure 1.6). In this way and unlike BR, CC is able to model the relationships among the labels without introducing more complexity. However, although it deals with the relationship among labels, it does not consider them, or any other characteristics of the data, at any moment of the building phase, such as to select the chain (which is randomly selected).

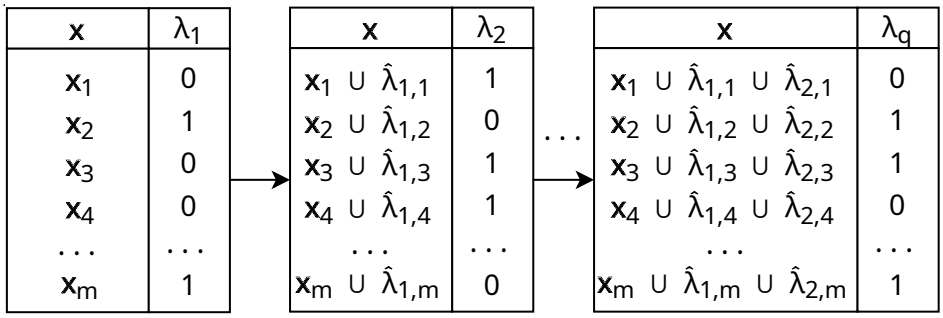


Figure 1.6: CC transformation.

Since the order of the chain has a determinant effect on its performance, other approaches have been proposed to select the best chain ordering, as Genetic Algorithm for ordering Classifier Chains (GACC) [37]. GACC uses a genetic algorithm to select the most appropriate chain for CC. Each individual in GACC represents a label permutation, i.e., a chain for CC, and they are evaluated using a linear combination of three evaluation metrics. In this way, while GACC looks for an optimal chain ordering, it also considers the relationship among labels to build the model.

Label Powerset (LP) [26] transforms the multi-label problem into a multi-class problem, creating one single-label dataset where each distinct labelset is considered as a different class, as seen in Figure 1.7. Then, LP uses any multi-class classification method to train a model with the new data, and the final prediction is obtained by transforming the predicted class to its corresponding labelset. LP considers all label correlations but its complexity is exponential with the number of labels. Furthermore, it is not able to predict a labelset that does not appear in the training dataset and, since many labelsets are usually associated with only few examples, it may lead to a highly imbalanced dataset which would make the learning process more difficult and less accurate. Therefore, LP is able to deal with the relationship among labels, but it increases the imbalance and the dimensionality of the output space, while it does not consider any of the characteristics when building the model.

$\mathbf{x}$	C
$\mathbf{x}_1$	C_{010}
$\mathbf{x}_2$	C_{101}
$\mathbf{x}_3$	C_{011}
$\mathbf{x}_4$	C_{010}
...	...
$\mathbf{x}_m$	C_{101}

Figure 1.7: LP transformation.

Pruned Sets (PS) [38] tries to reduce the complexity of LP, focusing on most important combinations of labels by pruning instances with less frequent labelsets. To compensate for this loss of information, it then reintroduces the pruned instances with a more frequent subset of labels. Thus, PS considers the imbalance of LP's

output space in its building phase to reduce its high dimensionality and complexity, although it might be still very complex in certain cases. Note that in Figure 1.8, labelsets appearing less than 2 times (as for instance #3) are pruned and reintroduced with a more frequent subsets.

x	C
x_1	C_{010}
x_2	C_{101}
x_3	C_{010}
x_4	C_{010}
...	...
x_m	C_{101}

Figure 1.8: PS transformation.

ChiDep [39] (a.k.a. LPBR) creates groups of dependent labels based on the χ^2 test for labels dependencies identification [40]. For each group of dependent labels it builds a LP classifier, while for each single label which is not in any group it builds a binary classifier. In the example in Figure 1.9, since labels λ_1 and λ_q are correlated they are modeled together with LP approach, while label λ_2 for example is modeled independently. ChiDep tries to reduce the disadvantages of the independence assumption of the binary methods and allows for simpler LP methods. Besides, ChiDep considers the relationship among group of labels and the dimensionality of the output space in building phase, therefore being able to reduce the imbalance in each model if the groups are small.

x	$C_{1,q}$
x_1	C_{00}
x_2	C_{11}
x_3	C_{01}
x_4	C_{00}
...	...
x_m	C_{11}

...

x	λ_2
x_1	1
x_2	0
x_3	1
x_4	1
...	...
x_m	0

Figure 1.9: ChiDep transformation.

Algorithm adaptation methods

Algorithm adaptation methods utilized almost all single-label classification techniques to directly handle multi-label data. Therefore, it is not necessary to transform the dataset.

Predictive Clustering Trees (PCTs) [41] are decision trees that can be viewed as a hierarchy of clusters; the root node of the PCT tree contains all data, and it is recursively partitioned into smaller clusters in children nodes. These trees are able to deal with multi-label data since the distance between two instances for the clustering algorithm is defined as the sum of Gini Indexes [42] of all labels; therefore it not only is able to model the relationship among labels but it also consider them in the building phase.

The well-known instance-based *k*-Nearest Neighbors (*k*NN) method has been also adapted to MLC. Multi-Label *k*-Nearest Neighbors (ML-*k*NN) [43] deals with multi-label data by finding the *k* nearest neighbors of a given instance, counting the number of neighbors belonging to each label, and using the *maximum a posteriori* principle to predict the labels for the given instance. As ML-*k*NN considers all label assignments of the *k*-nearest neighbors to label a new instance, it implicitly considers the relationship among labels to build the model. Besides, Instance-Based learning by Logistic Regression for Multi-Label classification (IBLR-ML) [44] is another example of adaptation of instance-based algorithms to MLC. It uses the labels of neighbor instances as extra input attributes in a logistic regression scheme. Therefore, it considers the relationships among labels at its building phase.

Neural networks were also adapted to MLC. Back-Propagation for Multi-Label Learning (BP-MLL) [29] defines a new error function taking into account the predicted ranking of labels. The ranking of labels also imply the relationship among labels, so BP-MLL considers them in the building phase. Other learning algorithms have been also proposed recently for deep neural networks in multi-label classification, such as the studies in [45] and [46].

A thorough description of MLC algorithm adaptation methods can be found in [22].

Ensembles of Multi-Label Classifiers

The third group of methods consists of the EMLCs. Although some methods such as BR or CC combine the predictions of several classifiers, only those that combine the predictions of several multi-label classifiers are considered as EMLCs.

Ensemble of BR classifiers (EBR) [36] builds an ensemble of n BR classifiers, each trained with a sample of the training dataset. The final prediction is obtained by combining the predictions (either bipartitions or confidences) of each of the members for each label independently. Generating an ensemble of BRs each with a random selection of instances provides diversity to the ensemble, therefore improving the performance of BR. However, EBR still does not take into account the relationship between labels.

Ensemble of Classifier Chains (ECC) [36] builds an ensemble of n CCs, each of them with a random chaining of labels and a random sample of m instances with replacement of the training set. Then, the final prediction is obtained by averaging the confidence values for each label. Finally, a threshold function is used to create a bipartition between relevant and irrelevant labels. The diversity in ECC is generated by selecting different random subsets of the instances in each CC, as well as selecting different random chains of labels. The selection of several different chains reduces the risk of selecting a bad chain which could lead to a bad performance, however, they are all created randomly and not based on any of the characteristics of the data.

Multi-Label Stacking (MLS) [47], also called 2BR, involves applying BR twice. MLS first trains q independent binary classifiers, one for each label. Then, it learns a second (or meta) level of binary models, taking as additional inputs the outputs of all the first level binary models. Thanks to the stacked predictions of previous BR classifiers, MLS is able to deal with the relationships among labels, but it does not consider the rest of characteristics, while the diversity is achieved by using different feature space in each classifier.

Hierarchy Of Multi-label classifiERs (HOMER) [48] is a method designed for domains with large number of labels. It transforms a multi-label classification prob-

lem into a tree-shaped hierarchy of simpler multi-label problems. At each node with more than one label, c children are created by distributing the labels among them with the balanced k-means method [48], making labels belonging to the same subset as similar as possible. Therefore, HOMER considers the relationship among labels to build the model, making it able to handle with smaller subsets of labels in each node, so that the dimensionality of the output space in each of them is reduced, also reducing the imbalance depending on the internal multi-label classifier used. The diversity in HOMER is generated by selecting a subset of the labels and also by filtering the instances in each classifier, keeping only those which are annotated with at least one label.

AdaBoost.MH [49] is an extension to multi-label learning of the extensively studied AdaBoost algorithm [50]. AdaBoost.MH not only maintains a set of weights over the instances as AdaBoost does, but also over the labels. Thus, training instances and their corresponding labels that are hard to predict, get incrementally higher weights in following classifiers, while instances and labels that are easier to classify get lower weights. The diversity in AdaBoost.MH is generated by using different weights for both instances and labels. However, it is not able to deal with any of the main problems of MLC.

Dynamic selection and Circulating Combination-based Clustering (D3C) [51] uses a dynamic ensemble method, where several single-label classifiers of different type are built for each label independently, and then uses clustering and dynamic selection to select a subset of accurate and diverse base methods for each of the labels. As it builds a set of binary classifiers, it does not deal with any of the main multi-label problems.

Ensemble of Pruned Sets (EPS) [52] builds an ensemble of n PSs where each classifier is trained with a sample of the training set without replacement. The predictions of each classifier are combined into a final prediction by a voting scheme using a prediction threshold t. The use of many PSs with different data subsets avoids overfitting effects of pruning instances, but as PS, in datasets with a high number of labels the complexity can be still very high. The diversity in EPS is created by the random selection of instances in each base classifier.

RAndom *k*-labELsets (RA*k*EL) [53] breaks the full set of labels into random small subsets of *k* labels (a.k.a. *k*-labelsets), then training a LP for each of the *k*-labelsets as base classifiers. Each model provides binary predictions for each label in its corresponding *k*-labelset, and these outputs are combined for a multi-label prediction following a majority voting process for each label. RA*k*EL is much simpler than LP since it only considers a small subset of labels at once, and also overcomes the problem of LP of not being able to predict a labelset that does not appear in the training dataset by means of voting.

In this way, RA*k*EL handles with the three main problems of the MLC: it is able to detect the compound dependencies among labels, it reduces the dimensionality of the output space by selecting small subsets of labels, and also the imbalance of each of the base models is not usually high since the reduced number of labels in each of them. However, it does not consider neither the relationship among labels nor the imbalance of the label space to select the *k*-labelsets, but it selects the *k*-labelsets just randomly, and does not guarantee neither that all labels are considered nor the number of times that each label appears in the ensemble.

Triple Random Ensemble for Multi-Label Classification (TREMLC) [54] is based on the random selection of features, labels and instances in each classifier of the ensemble. Then, a LP is built over each randomly selected data. The final prediction of TREMLC is obtained by majority voting. Therefore, TREMLC uses three ways to generate diversity in the ensemble, while dealing with main MLC problems in the same way than RA*k*EL does.

Ensemble of ChiDep classifiers (CDE) [39] is based on ChiDep. CDE first randomly generates a large number (e.g. 10,000) of possible label sets partitions. Then, a score for each partition is computed based on the χ^2 score for all label pairs in the partition. Finally, CDE selects the n distinct top scored partitions, generating a ChiDep model with each partition. For the classification of a new instance, a voting process with a threshold t is used to calculate the final prediction. CDE is able to deal with all three problems in MLC, as well as it considers the relationship among labels and the dimensionality of the output space when building the model. The diversity in CDE is generated by selecting a different partition on each classifier.

Random Forest of Predictive Clustering Trees (RF-PCT) [55] generates an ensemble which uses PCTs as base classifiers. As random forest [56], each base classifier of RF-PCT uses a different set of instances sampled with replacement, and also selects at each node of the tree the best feature from a random subset of the attributes. This double random selection over the instances and the features provides diversity to the base classifiers of the ensemble. For the prediction of a new instance, it averages the confidence values of all base classifiers for each label, and uses a threshold t to determine if the label is relevant or not.

Clustering-Based method for Multi-Label Classification (CBMLC) [57] involves two steps. In the first step, CBMLC groups the training data into c clusters only considering the features (not the labels). In the second step, it uses a multi-label algorithm to build a classifier over the data of each cluster, producing c multi-label classifiers. For the classification of an unknown instance, CBMLC first finds the closest cluster to the instance and then uses the corresponding classifier to classify it. By generating smaller problems, CBMLC is able to deal with the imbalance and the high dimensionality of the output space. Furthermore, it would be able to deal with the relationship among labels if a multi-label base method that models these relationships is used. On the other hand, CBMLC does not consider the labels when selecting the clusters, so it does not consider any of the characteristics of the data when building the model. Finally, CBMLC obtains diverse classifiers by the selection of instances and also labels in each cluster.

Ensemble of Multi-Label Sampling (EMLS) [58] builds an ensemble where each member is built over a random sample of the data using Multi-Label Synthetic Oversampling based on the Local distribution (MLSOL), which aims to deal with the imbalance of the data. Therefore, EMLS is able to deal with relationship among labels and it also deals with and considers the imbalance of the data in its building phase.

Given the advantages that ensemble methods have over individual ones in a wide range of machine learning tasks [1], including in multi-label classification [36, 53, 38], and also given the successful application of EMLCs to real-world problems [59, 60], we focused our research around EMLCs. A more comprehensive and detailed study of state-of-the-art EMLCs is further performed in Chapter 3.

1.2.4 Datasets characterization metrics

The fact that instances in multi-label datasets are associated with several labels simultaneously, leads to the need of defining new metrics to characterize the multi-label datasets that did not exist in traditional classification. The characterization of the data should be the first step to make before performing any machine learning technique. It is essential since help us to know what kind of data we are dealing with, therefore enabling to correctly preprocess the data or contributing to the selection of the method that best fits to it. In this section, we provide a summary of the most important metrics to characterize multi-label datasets, being those that we extensively use in this document. However, a wider study including a greater number of metrics is presented in [J2].

Let us remember that m, d, and q are the number of instances, features, and labels of the dataset. Unlike in single-label datasets, where the dimensionality of the data is usually related to the number of instances and/or input attributes, the dimensionality (*Dim*) of a multi-label dataset $\mathcal{D}$ is defined as the product of the number of instances, features, and labels (Equation 1.1) [61]. The greater the value, the more complex the dataset.

$$Dim(\mathcal{D}) = m \times d \times q \tag{1.1}$$

The cardinality (*Card*), defined in Equation 1.2, measures the mean number of labels associated to each instance [62]. High cardinality values mean that an instance is expected to have a greater number of labels associated; while lower values of cardinality means that each instance is associated with few labels.

$$Card(\mathcal{D}) = \frac{1}{m}\sum_{i=1}^{m}|Y_i| \tag{1.2}$$

While cardinality measures the average number of labels associated with each instance, it does not consider the total number of labels in the problem. For example, it should be interpreted differently to have a cardinality of 2 in problems with 5 labels than in problems with 100 labels. Therefore, density (*Dens*) is defined as

the cardinality divided by the total number of labels (Equation 1.3) [62]. Higher values of density means that a greater ratio of the possible labels are related with each instance.

$$Dens(\mathcal{D}) = \frac{Card(\mathcal{D})}{q} \tag{1.3}$$

The diversity (*Div*), which defined in Equation 1.4, represents the ratio of labelsets appearing in the dataset divided by the number of possible labelsets [62]. The possible number of labelsets in a dataset $\mathcal{D}$ usually is 2^q; however, if the number of instances m is lower than 2^q, the maximum number of possible labelsets in the dataset is m. The greater the value of diversity, the greater the number of combinations of labels with respect to the number of labels; so datasets with high diversity could be difficult to model with some approaches such as LP.

$$Div(\mathcal{D}) = \frac{\#Labelsets(\mathcal{D})}{\max(2^q, m)} \tag{1.4}$$

Given the imbalance problem in multi-label data, where some labels could be very frequent and other labels be barely present in the dataset, metrics to evaluate the imbalance of a multi-label dataset are defined. The average imbalance ratio (*avgIR*) measures how imbalanced are the labels in average for the whole dataset [63]. As seen in Equation 1.5, the *IR* is calculated for each label as the frequency of the most frequent label divided by the frequency of the current label.

$$avgIR(\mathcal{D}) = \frac{1}{q}\sum_{l=1}^{q}\frac{\arg\max_{\lambda' \in \mathcal{Y}}(f_{\lambda'})}{f_l} \tag{1.5}$$

Finally, as labels could be related to each other, metrics to evaluate the degree of relationship of the labels of the dataset are proposed. Coefficients Chi (χ^2) [40] and Phi (ϕ) [64] are usually used to identify the unconditionally relation between pairs of labels, and both are related (one of them could be calculated given the value of the other one). If value of χ^2 for a pair of labels is greater than 6.635, labels are considered dependent at 99% confidence [40]. On this basis, the ratio

of dependent label pairs by Chi-square test (*rDep*) is defined as the proportion of pairs of labels dependent at 99% confidence [65]. Equation 1.6 defines *rDep* metric, first calculating the number of dependent labels, and then averaging by the total number of label pairs.

$$rDep(\mathcal{D}) = \left(\sum_{l=1}^{q-1} \sum_{j=l+1}^{q} [\![\chi^2(\lambda_l, \lambda_j) > 6.635]\!] \right) \times \left(\frac{q(q-1)}{2} \right)^{-1} \tag{1.6}$$

1.2.5 Multi-label datasets

A great deal of benchmark datasets to assess multi-label classification methods have emerged in the last years. We have created a publicly available repository of multi-label datasets[1], which have been gathered from more than 30 sources. In Table 1.3 the multi-label datasets used in this document are presented, including their characteristics and references, and sorted in alphabetic order.

In each of the experiments performed in the present document, the datasets were selected according to their characteristics, in order to have a set of diverse datasets. Furthermore, in each of the experiments, the requirements in terms of characteristics of the datasets would be different to other. Finally, in some experiments, very complex datasets could not be used due to the high complexity of methods. As a consequence, only a subset of the datasets available in the repository are included in Table 1.3. Furthermore, note that datasets Mediamill*, Nus-Wide BoW*, and Tmc2007-500* were obtained by randomly selecting 5%, 1%, and 10% of the instances of the original dataset respectively.

1.2.6 Evaluation metrics

In multi-label classification, given that each instance is associated with several labels simultaneously, the predictions can be regarded as totally correct (all the predictions for all labels are correct), totally wrong (all the predictions are wrong), or partially correct, (only some of the relevant labels are predicted as relevant). As a consequence, many metrics have been proposed in the literature to evalu-

[1] https://www.uco.es/kdis/mllresources/

Table 1.3: Datasets and their characteristics, such as instances (*m*), attributes (*d*), labels (*q*), cardinality (*Card*), density (*Dens*), diversity (*Div*), average imbalance ratio (*agvIR*), ratio of dependent label pairs (*rDep*), and dimensionality (*Dim*).

Dataset	Domain	*m*	*d*	*q*	*Card*	*Dens*	*Div*	*avgIR*	*rDep*	*Dim*	Ref
20NG	Text	19300	1006	20	1.029	0.051	0.003	1.007	0.984	3.88E+08	[17]
3sources_bbc1000	Text	352	1000	6	1.125	0.188	0.234	1.718	0.733	2.11E+06	[67]
3sources_guardian1000	Text	302	1000	6	1.126	0.188	0.219	1.773	0.667	1.81E+06	[67]
3sources_inter3000	Text	169	3000	6	1.142	0.190	0.172	1.766	0.400	3.04E+06	[67]
3sources_reuters1000	Text	294	1000	6	1.126	0.188	0.219	1.789	0.667	1.76E+06	[67]
Birds	Audio	645	260	19	1.014	0.053	0.206	5.407	0.123	3.19E+06	[18]
CAL500	Music	502	68	174	26.044	0.150	1.000	20.578	0.192	5.94E+06	[68]
CHD_49	Medicine	555	49	6	2.580	0.430	0.531	5.766	0.267	1.63E+05	[69]
Emotions	Music	593	72	6	1.868	0.311	0.422	1.478	0.933	2.56E+05	[48]
Enron	Text	1702	1001	53	3.378	0.064	0.442	73.953	0.141	9.03E+07	[52]
EukaryotePseAAC	Biology	7766	440	22	1.146	0.052	0.014	45.012	0.281	7.52E+07	[70]
Flags	Image	194	19	7	3.392	0.485	0.422	2.255	0.381	2.58E+04	[37]
Genbase	Biology	662	1186	27	1.252	0.046	0.048	37.315	0.157	2.12E+07	[71]
GnegativePseAAC	Biology	1392	440	8	1.046	0.131	0.074	18.448	0.536	4.90E+06	[70]
HumanPseAAC	Biology	3106	440	14	1.185	0.085	0.027	15.289	0.418	1.91E+07	[70]
Langlog	Text	1460	1004	75	1.180	0.016	0.208	39.267	0.035	1.10E+08	[61]
Mediamill	Video	43910	120	101	4.376	0.043	0.149	256.405	0.342	5.32E+08	[72]
Mediamill*	Video	2195	120	101	4.430	0.044	0.393	294.599	0.116	2.80E+07	[72]
Medical	Text	978	1449	45	1.245	0.028	0.096	89.501	0.039	6.38E+07	[20]
Nus-Wide BoW*	Image	2696	501	81	1.863	0.023	0.302	89.130	0.087	2.80E+07	[73]
PlantGO	Biology	978	3091	12	1.079	0.090	0.033	6.690	0.318	3.63E+07	[70]
PlantPseAAC	Biology	978	440	12	1.079	0.090	0.033	6.690	0.318	5.16E+06	[70]
Scene	Image	2407	294	6	1.074	0.179	0.234	1.254	0.933	4.25E+06	[26]
Slashdot	Text	3782	1079	22	1.181	0.054	0.041	19.462	0.273	8.98E+07	[61]
Stackex_coffee	Text	225	1763	123	1.987	0.016	0.773	27.241	0.017	4.88E+07	[74]
Tmc2007-500*	Text	2860	500	22	2.230	0.101	0.136	17.225	0.364	3.15E+07	[75]
Water-quality	Chemistry	1060	16	14	5.073	0.362	0.778	1.767	0.473	2.37E+05	[76]
Yeast	Biology	2417	103	14	4.237	0.303	0.082	7.197	0.670	3.49E+06	[21]
Yelp	Text	10810	671	5	1.638	0.328	1.000	2.876	0.700	3.63E+07	[77]

ate multi-label predictions. These metrics are categorized into example-based and label-based metrics [66].

Example-based metrics

Example-based metrics calculate the metric for each instance of the data, and then, they average their value by the number of instances. They give the same weight to all instances, so they are not biased by the imbalance of the labels, which is one of the main problems in MLC.

Most of the example-based metrics (except Hamming loss) are considered as non-decomposable metrics [78]. The non-decomposable metrics evaluate the multi-label prediction as a whole, unlike others that evaluate the prediction for each label separately. So, non-decomposable metrics better capture the inherently relationship among labels when evaluating the predictions, instead of considering them independently.

Hamming loss (HL) computes the average number of times that each label is incorrectly predicted, including both prediction errors (an irrelevant labels was predicted as relevant) and omission errors (a relevant label was not predicted). As it measures the error in prediction, HL is to be minimized. Hereafter, it is indicated with $\downarrow$ if a metric is minimized and with $\uparrow$ if it is maximized. HL is defined in Equation 1.7, Δ being the symmetric difference between two binary sets. This is an often used evaluation metric in multi-label problems; however, in datasets with a high-dimensional output space and a low number of labels associated to each instance, HL tends to be zero in most cases. Therefore, in some problems, this metric should be cautiously used.

$$\downarrow \text{HL} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{q}|Y_i \Delta \hat{Y}_i| \tag{1.7}$$

Subset accuracy (SA) is a strict metric that measures the ratio of instances whose prediction is totally correct, including the correct predictions of all relevant and irrelevant labels. It is defined in Equation 1.8, where $[\![\pi]\!]$ returns 1 if predicate π is true and 0 otherwise. As HL, SA must be cautiously used. It is a very useful metric,

but, in problems where the output space is very large, the correct prediction of all labels could be very difficult and therefore, the fact of not considering partially correct predictions would lead to a non-accurate assessment of the methods.

$$\uparrow \text{SA} = \frac{1}{m}\sum_{i=1}^{m} [\![Y_i = \hat{Y}_i]\!] \tag{1.8}$$

On the other hand, evaluation metrics from traditional classification have been adapted as example-based metrics for MLC. Precision (ExP), defined in Equation 1.9, measures the ratio of correctly predicted labels from the total number of predicted labels. Recall (ExR), defined in Equation 1.10, measures the ratio of correctly predicted labels from all true labels. FMeasure (ExF), defined in Equation 1.11, computes the harmonic mean between precision and recall for each instance, thus considering both false positives (irrelevant labels predicted) and false negatives (non-predicted relevant labels) in its calculation. Specificity (ExS), defined in Equation 1.12, measures the ratio of irrelevant labels correctly predicted from the true set of irrelevant labels. Finally, Accuracy (ExAcc), defined in Equation 1.13, measures the ratio of correctly predicted labels to the total number of labels of the given instance, including true and predicted labels.

$$\uparrow \text{ExP} = \frac{1}{m}\sum_{i=1}^{m} \frac{\left|Y_i \cap \hat{Y}_i\right|}{\left|\hat{Y}_i\right|} \tag{1.9}$$

$$\uparrow \text{ExR} = \frac{1}{m}\sum_{i=1}^{m} \frac{\left|Y_i \cap \hat{Y}_i\right|}{|Y_i|} \tag{1.10}$$

$$\uparrow \text{ExF} = \frac{1}{m}\sum_{i=1}^{m} \frac{2 \times \left|Y_i \cap \hat{Y}_i\right|}{|Y_i| + \left|\hat{Y}_i\right|} \tag{1.11}$$

$$\uparrow \text{ExS} = \frac{1}{m}\sum_{i=1}^{m} \frac{\left|\overline{Y_i} \cap \overline{\hat{Y}_i}\right|}{\left|\overline{Y_i}\right|} \tag{1.12}$$

$$\uparrow \text{ExAcc} = \frac{1}{m}\sum_{i=1}^{m} \frac{\left|Y_i \cap \hat{Y}_i\right|}{\left|Y_i \cup \hat{Y}_i\right|} \tag{1.13}$$

Label-based metrics

Label-based evaluation metrics, different from example-based, are calculated according to the labels instead of instances. Any evaluation metric for single-label classification can be calculated in multi-label scenarios as label-based metric.

Label-based metrics, in turn, can be calculated by following two different approaches: micro-averaged and macro-averaged approaches. Let B be an evaluation metric for binary classification calculated based on the true positives (tp), false positives (fp), false negatives (fn), and true negatives (tn) of the confusion matrix (see Table 1.4). Micro-averaged metrics first join the confusion matrices of all labels and then calculate the metric (Equation 1.14), while macro-averaged calculate the metric for each label and then average their values for all labels (Equation 1.15). As a consequence, the former are more biased by more frequent labels, while the latter give the same importance to all labels in their calculation. Therefore, we have to cautiously select if micro or macro-averaged metrics are used depending on the problem at hand.

Table 1.4: Confusion matrix.

		True	
		Relevant	Irrelevant
Pred	Relevant	tp	fp
	Irrelevant	fn	tn

$$\text{MiB} = B\left(\sum_{l=1}^{q} tp_l, \sum_{l=1}^{q} fp_l, \sum_{l=1}^{q} fn_l, \sum_{l=1}^{q} tn_l\right) \tag{1.14}$$

$$\text{MaB} = \frac{1}{q}\sum_{l=1}^{q} B(tp_l, fp_l, fn_l, tn_l) \tag{1.15}$$

Following, precision, recall, FMeasure, specificity and accuracy metrics are defined in Equations 1.16 to 1.20 as micro-averaged metrics, and in Equations 1.21 to 1.25 as macro-averaged metrics, respectively.

$$\uparrow \text{MiP} = \frac{\sum_{l=1}^{q} tp_l}{\sum_{l=1}^{q} tp_l + \sum_{l=1}^{q} fp_l} \tag{1.16}$$

$$\uparrow \text{MiR} = \frac{\sum_{l=1}^{q} tp_l}{\sum_{l=1}^{q} tp_l + \sum_{l=1}^{q} fn_l} \tag{1.17}$$

$$\uparrow \text{MiF} = 2 \times \frac{\textit{MiP} \times \textit{MiR}}{\textit{MiP} + \textit{MiR}} \tag{1.18}$$

$$\uparrow \text{MiS} = \frac{\sum_{l=1}^{q} tn_l}{\sum_{l=1}^{q} tn_l + \sum_{l=1}^{q} fp_l} \tag{1.19}$$

$$\uparrow \text{MiAcc} = \frac{\sum_{l=1}^{q} tp_l + \sum_{l=1}^{q} tn_l}{\sum_{l=1}^{q} tp_l + \sum_{l=1}^{q} tn_l + \sum_{l=1}^{q} fp_l + \sum_{l=1}^{q} fn_l} \tag{1.20}$$

$$\uparrow \text{MaP} = \frac{1}{q} \sum_{l=1}^{q} \frac{tp_l}{tp_l + fp_l} \tag{1.21}$$

$$\uparrow \text{MaR} = \frac{1}{q} \sum_{l=1}^{q} \frac{tp_l}{tp_l + fn_l} \tag{1.22}$$

$$\uparrow \text{MaF} = 2 \times \frac{\textit{MaP} \times \textit{MaR}}{\textit{MaP} + \textit{MaR}} \tag{1.23}$$

$$\uparrow \text{MaS} = \frac{1}{q} \sum_{l=1}^{q} \frac{tn_l}{tn_l + fp_l} \tag{1.24}$$

$$\uparrow \text{MaAcc} = \frac{1}{q} \sum_{l=1}^{q} \frac{tp_l + tn_l}{tp_l + tn_l + fp_l + fn_l} \tag{1.25}$$

Pereira *et al.* carried out a study of correlation among evaluation metrics in MLC [79], in order to select a subset of metric to assess the performance of MLC algorithms that are independent among them. Based on this study, we conclude that HL, SA, MaP, and MaR are an interesting subset of evaluation metrics, almost independent of each other, to evaluate MLC methods in overall. Using, in addition other evaluation metrics such as MaF could be redundant, since it is a combination of MaP and MaR, which are already considered. In addition, usually the labels that are most important, interesting or difficult to predict are the minority labels,

so macro-averaged approach, which gives the same importance to all labels is preferred over micro-averaged approach. The study in [79] did not consider the MaS metric; however, we consider that measuring the ratio of correctly predicted irrelevant labels whould be also interesting.

1.2.7 Software tools for multi-labeled data

Due to the incremental interest in solving multi-label problems in the last decade, some software tools have been proposed in order to deal with multi-label data. Here, we distinguish among two types of software tools: tools for learning from multi-label data and tools for characterizing multi-label datasets. Usually, first type of tools also include some metrics to characterize the datasets, but it is not their real purpose and therefore they are not as complete as the others.

With regard to the software tools for learning from multi-label data, we mainly introduce three: Mulan, Meka, and Scikit-multilearn.

Mulan [80] is probably the most used multi-label learning library. It is developed in Java, built on top of the widely known data mining library Weka [81], and publicly available[2] under the GNU General Public License (GPL). Mulan includes a wide variety of methods that are not available in any other library. Furthermore, Mulan uses *.arff* dataset format, but also needs a *.xml* file to differentiate the labels[3].

Meka [82] is a GPL licensed Java library[4] aiming to expand Weka for multi-label learning. Unlike Mulan, Meka does not only provide a Java API but also a graphic interface, so users can build multi-label methods without the need of programming. However, the variety of methods in Meka is lower than in Mulan, but in includes a wrapper to execute Mulan's methods if necessary. The data format in Meka is slightly different to the one of Mulan; Meka only needs an *.arff* file in which header it includes the attributes that are labels.

Scikit-multilearn [83] is a library for multi-label learning built in top of Python's

[2] http://mulan.sourceforge.net/
[3] More information at https://www.uco.es/kdis/mllresources/#DatasetFormat
[4] https://waikato.github.io/meka/

scikit-learn library, and it is available under BSD license[5]. Scikit-multilearn is specialized on graph methods such as neural networks and deep learning methods for multi-label learning; however, it includes a low variety for the rest of types of methods. As for the data format, scikit-multilearn uses a self-defined format instead of classical *.arff* files.

On the other hand, we highlight two main tools for the characterization of multi-label datasets: mldr and MLDA.

Mldr [74] is a R package to analyze multi-label data, distributed under GPL license[6]. It includes the calculation of main characterization metrics and graphics for these characteristics, as well as the application of several data transformation, such as BR and LP. The mldr package not only includes an API in R but also provides a graphic interface to help any user to characterize its data independently of its knowledge of R programming language.

Finally, we also developed a tool built in Java that provides both an API and a user interface to characterize multi-label datasets, called MLDA [J2]. MLDA does not only include the characterization metrics that mldr provides, but also it includes a wider number of metrics. In addition, MLDA provides techniques to preprocess and partition the data, that were very useful in the development of this work. MLDA is also publicly available under GPL license[7]. More information about this tool is provided in Section 7.1.

1.3 Evolutionary algorithms

EAs are biology-inspired search algorithms that are commonly used in problems which are difficult to tackle with any other analytical methods [84]. EAs are very flexible methods that has been applied to many machine learning tasks, such as clustering [85], pattern mining [84], and feature selection [86]. Note that although suitable and very useful for a wide range of problems, EAs are stochastic procedures that cannot guarantee that the global optima is reached [87].

[5] http://scikit.ml
[6] https://github.com/fcharte/mldr
[7] https://github.com/i02momuj/MLDA

In EAs, there usually exist a population of individuals, where each of the individuals represents a full or partial candidate solution to the problem. Furthermore, each individual have a fitness value associated, meaning how well this solution solves the problem, and thus leading the evolution of the population towards fitter or more adapted individuals, aiming to obtain an optimal solution (or set of solutions).

Although a wide range of variations of EAs have been defined thorough the years, they usually rely on the same structure [88]:

Population The population is a set of individuals of (usually) fixed size, where each represents a full or partial candidate solution to the problem. The individuals in the population evolve, and those that provide a better solution to the problem usually have a higher chance to remain in the population. However, individuals in the population not only need to be fitted to the problem, but they also need to be somewhere different among them, thus representing different locations in the search space, and avoiding to stuck in local optima.

Fitness The fitness function represents the requirements that individuals should met. In other words, it is a procedure that assigns a quality value to each individual, depending on how well they solve the problem. It is used to lead the evolution towards optimal solutions.

Parents selection The role of parents selection operator is to select those individuals that will later produce offspring (i.e., new individuals). Usually, the selection of parents is made in a probabilistic way, where fitter individuals are more probable to be selected, thus aiming to improve the quality of new solutions. However, low-quality solutions usually also have a small chance to be selected, in order to maintain diversity in the population and not get stuck soon in a local optima.

Genetic operators Individuals interact with each other and are modified to generate offspring by means of genetic operators. Widely used genetic operators are crossover and mutation. Crossover operator combines genetic material of several individuals (usually two) in order to create new individuals that are

similar to each parent. On the other hand, mutation operator modify a single individual and it is usually more disruptive than crossover, since it is able to discover new genetic material that was no previously present in the population. Both types of operators are stochastic, i.e., are based on random choices to create offspring.

Population update At the end of each generation of the EA, the size of the population should remain the same. Therefore, the population is updated considering individuals from both the parents and offspring sets. Common techniques are to replace the whole parents set by the offspring, to maintain the best parent in the following population, to combine both sets and select most suitable or diverse individuals, etc. Together with the parents selection, these two procedures lead the population to improve in quality.

Stop criterion In order to stop the execution of the EA, one or several stop criteria must be used. Commonly used stop criteria are the number of generations in the evolution, the maximum number of evaluations of individuals, maximum number of generations in which the population has not improved, etc.

The basic steps of EAs are those presented in Figure 1.10. First, a population p of $popSize$ individuals is generated, usually randomly or following any heuristic. Then, individuals in p are evaluated using the fitness function. Later, until the stop criteria is reached, parents are selected, usually based on their fitness value, where fitter individuals have more chance to be selected; genetic operators are applied to the selected individuals; offspring individuals in s are evaluated; and the population is updated considering individuals in both p and s sets. Finally, the population is usually returned.

In Figure 1.11 the operation of EAs is shown with a simple example of maximization of a one-dimensional function. In the figures, the x-axis represents the different positions in the search space, while the y-axis represents the fitness of each individual (i.e., the value of the function to maximize). As observed, at the beginning of the evolution, individuals are usually randomly created, so they are distributed throughout the search space (Figure 1.11a). After some generations,

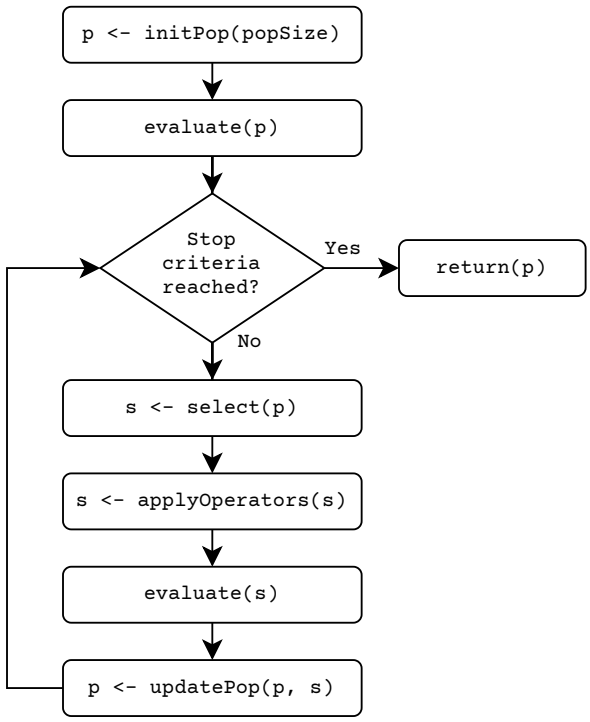
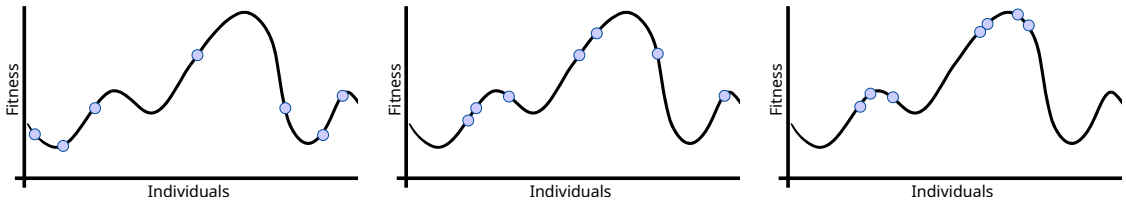


Figure 1.10: Main steps of EAs.

given the selection and use of genetic operators, individuals in regions with lower value of fitness tend to disappear, while they start to climb the hills in pursuit of more promising zones (Figure 1.11b). Finally, over the end of the EA, individuals would be gathered around optimal zones (Figure 1.11c). These individuals may be spread over several hills, but they could also be concentrated in a suboptimal zone, thus not reaching the global optimum. For this reason, it is essential to have both exploration (creating individuals in new zones of the search space) and exploitation (concentration and improvement of individuals in promising zones) mechanisms in the EA. Nevertheless, it should be fine-tuned the trade-off between exploration and exploitation in order to not to lead to a premature convergence, i.e., getting trapped in a local optimum.



(a) Beginning (b) Halfway (c) End

Figure 1.11: Operation of EAs along the iterations.

Based on this structure, a wide range of different evolutionary techniques have been proposed so far, such as Genetic Algorithms (GAs) [89], Genetic Programming (GP) [90], Gene Expression Programming (GEP) [91], Cooperative CoEvolutionary Algorithms (CCEAs) [92], or Particle Swarm Optimization (PSO) [93], among a large list of different frameworks.

In order to implement the EAs for our research, we have used the JCLEC library [94]. JCLEC is a software system for evolutionary computation, developed in Java, and publicly available[8] under GPL License. It provides a high-level software framework to do any kind of EA, and gives support for several predefined algorithms such as GAs and GP.

[8] http://jclec.sourceforge.net/

Chapter 2

Motivation and objectives

In this chapter, we present the motivation and main objectives of this dissertation.

A large number of EMLCs were proposed in the literature, and they have been proven to significantly outperform simpler methods [36, 53, 38]. However, as presented in Section 1.2.3, despite their promising performance most of them base the generation of diversity in the ensemble by randomly selecting subsets of features, instances, or labels. Furthermore, the characteristics of the data such as the relationship among label, the imbalance, and the high dimensionality of the output space are not considered, while these characteristics could provide useful tips in the building phase of the ensemble.

Thus, we consider that there is a need of proposing novel EMLC methods that take into account these characteristics, aiming to improve the state-of-the-art MLC methods proposed up to date. Besides that, the process of building an EMLC is defined as a search problem, where the aim is to obtain an optimal configuration for the EMLC. For this purpose EAs provide an optimal framework given the complex nature of the problem.

In the following, the main objectives (O1 - O5) of this Ph.D. thesis are presented. Furthermore, Table 2.1 provides a summary of the objectives and which of the three main research papers associated with this thesis satisfies the objectives. Full references of these papers are provided in Vita chapter.

Table 2.1: Objectives of the Ph.D. thesis and research papers that satisfy them.

	O1	O2	O3	O4	O5
[J4]	×				×
[J5]		×	×	×	×
[J6]		×	×	×	×

O1. Thorough MLC experimental review. We first aim to conduct a experimental review on MLC methods, with a special focus on EMLCs. The study of MLC methods has several sub-objectives associated with it: I) obtain a broad background on multi-label classification and ensemble methods, necessary to conduct the rest of our research; II) categorize existing EMLCs according to their characteristics, such as the multi-label classifier in which they are based and the way they generate diversity in the ensemble; III) perform an experimental study to conclude what method or family of methods perform better depending on the characteristics of the data (such as the degree of relationship among labels, the imbalance of the output space, or its dimensionality); and IV) give some tips to select the best EMLC for a given problem according to the characteristics of the data.

As a result of this study, we have published a journal paper [J4], which is presented in Chapter 3.

O2. Propose novel methods to build EMLCs. Given the lack of methods to build EMLCs considering the characteristics of the multi-labeled data, we aim to develop some algorithms, based on EAs, to build EMLCs while considering these characteristics. EAs have been widely used in optimization problems, so we use them with the goal of obtaining an optimal structure for the EMLC.

First, we propose an algorithm, called EME [J5], which is presented in Chapter 4. In EME, each individual in the EA represents an entire EMLC, where each of its members is a multi-label classifier focused on small subsets of the labels. Furthermore, we propose to use a multi-label method that strongly considers the relationship among labels (such as LP), so each member of the

ensemble is able to model these relationships but with relative low complexity. The fitness function is defined as a combination of the ExF and a coverage metric which ensures that all labels are equally present in the ensemble. We also propose a novel mutation operator that looks for combinations of more related labels, trying to improve the performance of multi-label classifiers.

Then, we propose a second evolutionary approach to build EMLCs, called EAGLET [J6], and presented in Chapter 5. Given the promising results of EME we developed a new approach, where each individual is a multi-label classifier based on subsets of the labels, instead of being the entire ensemble. One of the aims of EAGLET is to obtain a more efficient method than EME by evolving simpler individuals. On the other hand, EAGLET also aims to lead the evolution around highly accurate individuals that are then combined into an ensemble. However, not only accurate but also diverse individuals are needed to build a promising ensemble, so a process to ensure that diverse individuals are considered in the population is introduced, also ensuring to cover all labels in the population. Furthermore, as individuals are not ensembles but possible members of an ensemble, in each iteration the more promising individuals are used to generate an ensemble. For that, the individual with better predictive performance is first selected, and then, individuals that maximize both its predictive performance and the diversity of the current ensemble are selected to contribute to the ensemble, thus leading to select diverse individuals with promising predictive performance.

Finally, a preliminary version of other evolutionary approaches have been developed, such as a CCEA to build EMLCs [C13] (see Section 7.2), and a Grammar-Guided Genetic Programming (G3P) method to build tree-shaped EMLCs [C14] (see Section 7.3). Both have been accepted in international conferences.

O3. Comparison of proposed methods with the state-of-the-art. We aim to compare our evolutionary approaches with other state-of-the-art algorithms to build EMLCs, thus demonstrating their promising performance. The experimental studies carried out demonstrate, based on statistical tests, that both

EME and EAGLET significantly outperform other state-of-the-art MLC methods, including widely used EMLCs. Furthermore, EAGLET demonstrates to be far more efficient than EME, as expected.

O4. Application of developed models to real-world benchmarks. The developed models have been tested in up to sixteen real-world datasets covering a wide range of domains, such as multimedia annotation (both audio and images), biology (as yeast gene annotation, and predicting subcellular locations of proteins sequences in humans, plants, and bacterias), chemistry (as predicting the quality of water in rivers), and text categorization (including texts related to news, clinical free text reports, and forums).

O5. Integration of the proposed methods in relevant software platforms for distribution. In order to not only facilitate the reproducibility of the experimental results, but also to enrich the contribution to the scientific community, the proposed methods and other useful tools are integrated in software platforms (such in a GitHub repository) under the GPLv3 License[1]. In this way, any researcher interested in using our proposal, is able to use and/or to modify it. Following, the list of repositories associated with this thesis is provided.

ExecuteMulan. Library implemented to execute Mulan methods from command line. It has been used to carry out all the experiments in this thesis.

URL: https://github.com/kdis-lab/ExecuteMulan

EME. Repository including the code of the first evolutionary approach to build EMLCs, EME. It is written in Java, and developed using Mulan, Weka, and JCLEC libraries.

URL: https://github.com/kdis-lab/EME

EAGLET. Repository including the code of the second evolutionary approach to build EMLCs, EAGLET. It is written in Java, and developed using Mulan, Weka, and JCLEC libraries.

URL: https://github.com/kdis-lab/EAGLET

[1]The GNU General Public License (GPL) is a free, copyleft license for software and other kinds of works.

The rest of the document is organized as follows. Chapter 3 presents the literature review, categorization, and experimental comparison of EMLCs. Chapters 4 and 5 propose the two evolutionary approaches to build EMLCs, EME and EAGLET, respectively. Chapter 6 discusses the results obtained throughout the thesis, as well as concludes and presents lines of future work. Chapter 7 presents other publications associated with this Ph.D. thesis. Finally, full list of references and Vita of Ph.D. candidate are included.

Chapter 3

Experimental review and categorization of EMLCs

The main topic of this work is the development of ensemble methods for MLC, and in this chapter we provide a background of EMLCs. This background will provide knowledge of the previous work, as well as a starting point for developing new models.

In the literature, two studies were performed carrying out an experimental comparison of MLC methods [33, 95]. However, none of them included the state-of-the-art EMLCs. Therefore, given the absence of studies comparing the performance of state-of-the-art EMLCs, in this chapter we aim to provide a thorough definition of state-of-the-art EMLCs as well as a comparison and analysis of their performance.

While overviewing the state-of-the-art EMLCs, we found some common points by which EMLCs could be grouped or categorized. As no taxonomies were previously proposed in the literature to specifically categorize EMLCs, in this chapter we propose a novel taxonomy, which categorizes EMLCs according to two different criteria. First, the EMLCs are categorized depending on which multi-label classifier they are based, such as BR, LP, PCT, or maybe independent of the classifier. On the other hand, methods are also categorized regarding how they create a diverse ensemble, e.g., by dealing with different classifiers (or using different hyper-parameters of the same classifier), different subsets of labels, different subsets of the input attributes, or different subsets of instances in each of the base classifiers. This second criterion is based on the taxonomy proposed in [96]. Note that in the

first criterion, each EMLC can only be categorized into one of the groups; however, groups in the second criterion are not mutually exclusive, so an ensemble could appear in several groups simultaneously.

Once categorized within the taxonomy, we also perform an experimental study to compare the EMLCs, using twenty multi-label datasets and seven evaluation metrics in total. We divide the study into four different experiments. First, we evaluate the performance of the EMLCs depending on the imbalance of the datasets. For this purpose, we divide the datasets into little, moderately, and very imbalanced datasets, according to their *avgIR*. They are evaluated using MiF and MaF metrics, which compute the FMeasure from different points of view: the former giving more weight to more frequent labels, and the latter giving the same weight to all of them, regardless of their frequency.

Second, we evaluate the performance of the EMLCs depending on the relationship among labels. In this case, we divide the datasets into low, medium, and highly dependent, according to their *rDep*. In this experiment the EMLCs are evaluated using ExF and ExAcc, since they are non-decomposable metrics and evaluate the multi-label prediction as a whole, thus being able to capture the relationship among labels in the evaluation.

Third, we carry out a study on the efficiency of the EMLCs, for which we split the datasets according to their dimensionality, into small, medium, and large datasets. Their runtime in both training and testing phases is evaluated. In addition to the runtime, in this experiment we also use the HL to asses the predictive performance of EMLCs, thus being able to monitor these methods that are very fast but their performance is poor.

And fourth, we evaluate the EMLCs in general, using all twenty datasets together and the seven evaluation metrics used so far. Thus, we aim to provide a general view of how the different EMLCs perform regardless of the characteristics of the problem, and to give a ranking of methods based on their overall performance.

Finally, and as a consequence of previous study, we also provide some tips to select the best EMLC according to the characteristics of the data. In this way, we aim to provide useful knowledge to further researcher in multi-label classification,

helping them to select the method (or group of methods) that could perform best in their specific problem.

The experimental study proves that in scenarios with moderately and high labels imbalance, RA*k*EL is the most suitable method, since it builds base classifiers over lower imbalanced data while still considering the dependencies among labels. Furthermore, both RA*k*EL and ECC perform well when the labels are not much dependent among them; however, the Ensemble of Label Powersets (ELP) performs better when labels are more correlated among them, since it is able to model these dependencies in a stronger way. As for the efficiency, methods such as CBMLC are very efficient, while its performance is poor; on the other hand, CDE is so computationally complex that it did not finish its execution in some cases, and ECC, although obtaining a great predictive performance, is the second more complex method. Finally, ECC is the EMLC that achieves a better overall performance (without considering runtime), followed by RA*k*EL, and it is worth mentioning EPS, which is a great combination of both good performance and fast algorithm. Tables with full results of all the experimental studies are available to facilitate further comparisons[1].

Following, we present the journal paper associated with this chapter of the thesis [J4].

[1] https://www.uco.es/kdis/emlcreview/

Title	Review of ensembles of multi-label classifiers: Models, experimental study and prospects
Authors	Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastián Ventura
Journal	Information Fusion
Volume	44
Pages	33 - 45
Year	2018
DOI	10.1016/j.inffus.2017.12.001

IF (JCR 2018)	10.716
Category	Computer Science - Artificial Intelligence
Position	3/133 (Q1)
Cites (27/05/2020)	16 (WoS), 22 (Scopus), 35 (Google Scholar)

Information Fusion 44 (2018) 33–45

Contents lists available at ScienceDirect

Information Fusion

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

Review of ensembles of multi-label classifiers: Models, experimental study and prospects

Jose M. Moyano[a], Eva L. Gibaja[a], Krzysztof J. Cios[b,c], Sebastián Ventura[*,a,d,e]

[a] Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain
[b] Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284, USA
[c] Polish Academy of Sciences, Institute of Theoretical and Applied Informatics, Gliwice, Poland
[d] Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
[e] Knowledge Discovery and Intelligent Systems in Biomedicine Laboratory, Maimonides Biomedical Research Institute of Córdoba, Spain

ARTICLE INFO

Keywords:
Multi-label classification
Ensemble methods

ABSTRACT

The great attention given by the scientific community to multi-label learning in recent years has led to the development of a large number of methods, many of them based on ensembles. A comparison of the state-of-the-art in ensembles of multi-label classifiers over a wide set of 20 datasets have been carried out in this paper, evaluating their performance based on the characteristics of the datasets such as imbalance, dependence among labels and dimensionality. In each case, suggestions are given to choose the algorithm that fits best. Further, given the absence of taxonomies of ensembles of multi-label classifiers, a novel taxonomy for these methods is proposed.

1. Introduction

Ensemble learning combine individual learners from heterogeneous or homogeneous modeling in order to obtain a combined learner that improves the generalization ability and reduces the overfitting risk of each one of them [1,2]. In [3], Dietterich specified three reasons why an ensemble classifier is better than a single classifier. First, when picking only one single classifier, we run the risk of choosing a bad one. Second, many learning algorithms use local search and may not find the optimal classifier, so running several times the learning algorithm and combining the obtained models may result in a better approximation to the optimal classifier than any single one. Third, since in most machine learning problems the optimal function cannot be found, the optimal classifier may be reached by combining several classifiers. Ensemble methods have been successfully applied in many fields such as finance [4], bioinformatics [5], medicine [6], image retrieval [7] and recommender systems [8].

The development of ensemble models has been used in a large number of machine learning tasks, such as in Multi-Label Learning (MLL) and more specifically in Multi-Label Classification (MLC), where each object may have multiple labels associated with it [9,10]. Despite the fact that in MLC many algorithms are based on combining several classifiers, only those whose combine several classification methods that are able to deal with multi-label data are considered as Ensembles of Multi-Label Classifiers (EMLCs) [11]. EMLCs have been successfully applied in image retrieval [12] and predicting drug resistance [13].

Given the advantages of ensemble methods over simple methods, it is interesting to perform an experimental study of the state-of-the-art EMLCs. In [11] and [14] two experimental studies of MLC algorithms were performed. However, none of them includes the state-of-the-art EMLCs. Given the absence of experimental studies contemplating the state-of-the-art and the special characteristics of the EMLCs, the objective of this paper is to perform an experimental comparison and analysis of the state-of-the-art EMLCs over a range of datasets and evaluation metrics. This study is performed taking into account the characteristics of the data, such as imbalance, relationship among labels or dimensionality, indicating which EMLC achieves better performance in each case and giving some guidelines to select the best algorithm according to the characteristics of the dataset.

The rest of the paper is organized as follows: Section 2 presents background in MLC, Section 3 describes different EMLC methods and categorizes them into a novel taxonomy, Section 4 shows the experimental design, Section 5 describes and discusses the results of the experiments and also gives some guidelines to select the best EMLC according to the characteristics of the dataset and finally Section 6 presents conclusions of this work.

* Corresponding author.
E-mail address: sventura@uco.es (S. Ventura).

https://doi.org/10.1016/j.inffus.2017.12.001
Received 19 November 2017; Received in revised form 13 December 2017; Accepted 16 December 2017
Available online 18 December 2017

2. Background

In this section the formal definition of MLC and the main categorization of MLC methods are presented.

2.1. Formal definition of multi-label classification

Given a d-dimensional input space $\mathscr{X} = X_1 \times \cdots \times X_d$ and an output space of q labels $\mathscr{Y} = \{\lambda_1, \lambda_2, ..., \lambda_q\}$, $q > 1$, being the cardinality of each label $|\lambda_i| = 2$, a multi-label example can be defined as a pair $(\mathbf{x}, Y)$ where $\mathbf{x} = (x_1, \cdots, x_d) \in \mathscr{X}$ and $Y \subseteq \mathscr{Y}$ is called labelset. $\mathscr{D} = \{(\mathbf{x}_i, Y_i) | 1 \leq i \leq m\}$ is a multi-label dataset composed of a set of m instances [15].

The goal of multi-label classification is to construct a predictive model $h: \mathscr{X} \rightarrow 2^{\mathscr{Y}}$ which would provide a set of relevant labels for an unknown instance. Each instance may have several labels associated with it from the previously defined set of labels. So, for each $\mathbf{x} \in \mathscr{X}$, we have a bipartition $(Y, \overline{Y})$ of the label space $\mathscr{Y}$, where $Y = h(\mathbf{x})$ is the set of relevant labels and $\overline{Y}$ the set of irrelevant ones.

2.2. Multi-label classification algorithms

MLC algorithms are categorized into three main groups: problem transformation, algorithm adaptation and EMLCs [11]. Problem transformation methods transform a multi-label problem into one or several single-label problems, as Binary Relevance (BR) [16] that decomposes the multi-label learning problem into q independent binary classification problems, and Label Powerset (LP) [17], which generates a single-label dataset where each distinct labelset is considered as a different class. A more extensive list of problem transformation methods can be found in [15].

Algorithm adaptation methods adapted almost all classification techniques for multi-label learning, such as decision trees [18,19], support vector machines [20,21], neural networks [22,23] and instance-based algorithms [24,25].

Finally, the third group of MLC methods includes the EMLCs. There are many algorithms in multi-label classification, such as BR, that involve combination of several classifiers instead of a single classifier. However, in MLC only are considered as EMLCs those ensembles that involve the combination of several classification methods which are able to deal with multi-label data, so algorithms such as BR are not considered as EMLC since they combine several single-label methods. EMLCs are the object of this work and are described in detail in Section 3.

3. Ensembles of MLC

In this section a total of 16 state-of-the-art ensemble methods for MLC are described. They are first categorized depending on the method they are based. Once the methods have been described, they are then categorized based on a proposed taxonomy.

3.1. Ensembles based on Binary Relevance (BR)

BR combines predictions of several binary classifiers, one for each label in the multi-label dataset [16]. BR is simple, intuitive and resistant to overfitting label combinations because it does not take into account predefined combinations of labels. Thus, it can handle irregular labeling and is able to predict combinations of labels that did not appear in the original training set. However, BR's weakness is the fact that it does not take into account the relationship among the labels, assuming that the labels are independent while in most cases they are not. Several ensemble methods, which are defined below, were developed to overcome BR's problem.

3.1.1. Ensemble of Binary Relevance classifiers (EBR)

The Ensemble of Binary Relevance classifiers (EBR) [26] is generated using bagging [27] for each BR classifier. Generating an ensemble of BRs each with a random selection of instances improve performance of BR due to the diversity among base classifiers. However, EBR still does not take into account the relationship between labels.

3.1.2. Ensemble of Classifier Chains (ECC)

Classifier Chains (CC) [28] generate a chain of q binary datasets, where the feature space of each classifier is augmented with the label predictions of previous classifiers. The order of the selected chain has a direct impact on performance of the classifier, due to the error propagation along the chain at classification time when some of the classifiers in the chain predict poorly. To overcome this problem, Ensemble of Classifier Chains (ECC) [26] trains n CC, each one with a random chain built over a random selection of m training instances sampled with replacement. Then, the final prediction is obtained by averaging the confidence values for each label. Finally, a threshold function is used to create a bipartition of relevant and irrelevant labels.

Both CC and ECC pass label information among binary classifiers, so they take into account correlations among labels and overcome the problem of BR, which ignores such correlations. Also, using ECC reduces the risk of selecting a bad chain ordering which can lead to a bad prediction performance of the classifier. The diversity in ECC is generated by using different chains and by selecting random subsets of instances.

3.1.3. Multi-Label Stacking (MLS)

Multi-Label Stacking (MLS) [29], also called 2BR, involves applying BR twice. MLS first trains q independent binary classifiers, one for each label. Then, it learns a second (or meta) level of binary models, taking as additional inputs the outputs of all the first level binary models, thus taking into account the relationship among labels in the meta-level.

There exist several approaches of MLS depending on the way they gather the predictions in the base-level. $\text{MLS}_{\text{train}}$ uses the full training set for both base and meta levels, but this can lead to biased meta-level training data. MLS_{cv} partitions the data into F disjoints parts, generating each base-level classifier F times, each using $F-1$ partitions for training and the remaining for gathering the predictions. In this way it obtains a non-biased meta-level training set. However, this method is much slower than $\text{MLS}_{\text{train}}$. Another one, MLS_{ϕ} tries to not introduce irrelevant information into the meta-level. If a label is completely uncorrelated with the one being modeled, including its predicted value in the meta-level classifier introduces non interesting information and noise, which could lead to a worse performance. For that, MLS_{ϕ} uses the ϕ correlation coefficient [30] to determine if two labels are correlated or not, pruning labels that are not correlated with the one being modeled in each meta-level classifier.

In all MLS variants, the diversity of the ensemble is achieved by using different feature space in each classifier.

3.1.4. Hierarchy Of Multi-label classifiERs (HOMER)

Hierarchy Of Multi-label classifiERs (HOMER) [31] is a method designed for domains with large number of labels. It transform a multi-label classification problem into a tree-shaped hierarchy of simpler multi-label problems. At each node with more than one label, c children are created by distributing the labels among them with the balanced k-means method [31], making labels belonging to the same subset as similar as possible. To classify a new instance, HOMER starts with the root classifier and passes the instance to each child only if the parent predicted any of its labels. The union of the predicted labels by the leaves generates output for the given instance.

HOMER is based on BR since in each node a binary classifier is used, which predicts if any or none of the labels in the node are associated with the instance. The diversity in HOMER is generated by selecting a subset of the labels and also by filtering the instances in each classifier,

keeping only those which are annotated with at least one label.

3.1.5. AdaBoost.MH

AdaBoost algorithm [32] was extensively studied and used in several machine learning tasks [33–35]. The AdaBoost.MH [36] method not only maintains a set of weights over the instances as AdaBoost does, but also over the labels. Thus, training instances and their corresponding labels that are hard to predict, get incrementally higher weights in following classifiers while instances and labels that are easy to classify get lower weights. The diversity in AdaBoost.MH is generated by using different weights for both instances and labels.

AdaBoost.MH is based on BR since each instance is passed to q binary classifiers, so it is the same as applying AdaBoost to q binary classifiers [36].

3.2. Ensembles based on Label Powerset (LP)

LP generates a single-label dataset with a different class for each different combination of labels. In this way, LP takes into account label correlations but its complexity is exponential with the number of labels and it is not able to predict a labelset which does not appear in the training dataset. Besides, many labelsets are usually associated with only few examples, which may lead to an imbalanced dataset and make the learning process more difficult. Many multi-label ensemble classifiers, described below, have been proposed to overcome these disadvantages of LP.

3.2.1. Ensemble of Label Powerset classifiers (ELP)

Ensemble of Label Powerset (ELP) classifiers is proposed in this paper to compare it as a baseline with other LP-based methods. It uses bagging to generate diversity of classifiers and then combines the predictions of the base classifiers by majority voting. As it combines several LP predictions, ELP is able to predict a labelset that does not appear in the original training set. However, its complexity is not reduced with respect to LP as well as the problem of imbalance is not solved.

3.2.2. Ensemble of Pruned Sets (EPS)

Pruned Sets (PS) [37] is similar to LP but it focuses on the most important relationships of labels by pruning the infrequently occurring labelsets, reducing the complexity of the algorithm.

Ensemble of Pruned Sets (EPS) was proposed in [37] to prevent from overfitting effects of pruning and to allow predicting labelsets that do not appear in train data. EPS trains n PS models, each over a subset without replacement of the instances of the original training set. The predictions of each classifier are combined into a final prediction by a voting scheme using a prediction threshold t. The diversity in EPS is created by the random data selected for each base classifier.

3.2.3. RAndom k-labELsets (RAkEL)

RAndom k-labELsets (RAkEL) [38] randomly breaks the set of labels into several small-sized labelsets. RAkEL selects n random k-labelsets and learns n LP classifiers, each one focusing on its own k-labelset. Each model provides binary predictions for each label in its corresponding k-labelset. These outputs are combined for a multi-label prediction following a majority voting process for each label.

RAkEL has several advantages over LP. First, the LP tasks of each classifier are much simpler since they only consider a small subset of the labels. Also, the base classifiers include a much more balanced distribution of classes than using LP with the full set of labels. Further, RAkEL allows to predict a labelset that does not appear in the original training set.

A variation of RAkEL, called RAkEL++ [39], uses the confidence values of each classifier instead of bipartitions in order to generate the final prediction for each label. Another variation, called RAkELd [38], generates disjoint subsets of k labels, taking into account each label exactly once and reducing the complexity of other RAkEL variants

The diversity in all the variants of RAkEL is generated by different selection of labels in each classifier.

3.2.4. Triple Random Ensemble for Multi-Label Classification (TREMLC)

Triple Random Ensemble for Multi-Label Classification (TREMLC) [40] is based on the random selection of features, labels and instances in each classifier of the ensemble. In this way, TREMLC uses three ways to obtain diversity of base classifiers. Then, a LP is built over each randomly selected data. The final prediction of TREMLC is obtained by majority voting.

3.2.5. Chi-Dep Ensemble (CDE)

Chi-Dep [41] groups dependent labels by χ^2 score [42], building a LP classifier for each group of dependent labels and a binary classifier for each independent label. In this way, it achieves an optimal trade-off between simple (single-label) and complex (LP) models respectively, reducing the disadvantages of both approaches.

Based on the Chi-Dep algorithm, an Ensemble of Chi-Dep classifiers (CDE) was proposed in [43]. CDE first randomly generates a large number (e.g., 10000) of possible label sets partitions. Then, a score for each partition is computed based on the χ^2 score for all label pairs in the partition. Finally, CDE selects the n distinct top scored partitions, generating a Chi-Dep algorithm with each partition. For the classification of a new instance, a voting process with a threshold t is used to calculate the final prediction.

The diversity in CDE is generated by selecting a different partition on each classifier.

3.3. Ensembles based on Predictive Clustering Trees: Random Forest of Predictive Clustering Trees (RF-PCT)

Predictive Clustering Trees (PCTs) [19] are decision trees that can be viewed as a hierarchy of clusters in such a way that the intra-cluster variation is minimized. The root node of PCT contains all data, and it is recursively partitioned into smaller clusters in children nodes. In order to construct a PCT in MLC, the distance between two instances is usually computed as the sum of Gini Indices [44] of the labels.

The Random Forest of Predictive Clustering Trees (RF-PCT) [45] generates an ensemble which uses PCTs as base classifiers. As random forest [46], each base classifier of RF-PCT uses a different set of instances sampled by bagging, and also selects at each node of the tree the best feature from a random subset of the attributes. This double random selection over the instances and the features provides diversity to the base classifiers of the ensemble.

For the prediction of a new instance, it averages the confidence values of all base classifiers for each label, and uses a threshold t to determine if the label is relevant or not.

3.4. Ensembles independent of base classifiers: Clustering-Based for Multi-Label Classification (CBMLC)

The previously described EMLCs were designed based on a specific multi-label method. However, there also exist EMLCs which are completely independent of the multi-label classifier used.

Clustering-Based method for Multi-Label Classification (CBMLC) [47] has two steps. In the first step, CBMLC groups the training data into c clusters using a clustering algorithm and only considers the features (not the labels). It is expected that similar objects are associated with similar labels, which results in a reduced label space in each of the classifiers. This may improve the predictive performance of each classifier, as well as to reduce the training and testing time. In the second step, it uses the multi-label algorithm to build a classifier over the data of each cluster, producing c multi-label classifiers. For the classification of an unknown instance, CBMLC first finds the cluster closest to the instance and then uses the corresponding classifier to classify it. LP was used as multi-label classifier and k-means [48] was

Table 1
EMLC methods. State-of-the-art ensemble of MLC methods are categorized depending on the method they are based (BR, LP, PCT or independent) and the way the diversity of the ensemble is obtained as A (*classifier level*), B (*label level*), C (*feature level*) or D (*data level*).

Abbreviation	Method name	Level				Reference	Year
		A	B	C	D		
	Ensembles based on BR						
EBR	Ensemble of Binary Relevance classifiers (bagging)				•	[26]	2011
ECC	Ensemble of Classifier Chains	•			•	[26]	2011
MLS_{train}	Multi-Label Stacking using train data for meta-level			•		[29]	2009
MLS_{cv}	Multi-Label Stacking using cv for meta-level			•		[29]	2009
MLS_{ϕ}	Multi-Label Stacking pruning meta-level			•		[29]	2009
HOMER	Hierarchy Of Multi-label classifiERs		•		•	[31]	2008
AdaBoost.MH	AdaBoost.MH		•		•	[36]	2000
	Ensembles based on LP						
ELP	Ensemble of Label Powerset classifiers (bagging)				•	–	–
EPS	Ensemble of Pruned Sets				•	[37]	2008
RA*k*EL	Random *k*-labELsets		•			[38]	2011
RA*k*EL++	Random *k*-labELsets using confidences		•			[39]	2014
RA*k*ELd	Random *k*-labELsets with disjoint labelsets		•			[38]	2011
TREMLC	Triple Random Ensemble for MLC		•	•	•	[40]	2010
CDE	Chi-Dep Ensemble	•				[43]	2010
	Ensembles based on PCT						
RF-PCT	Random Forest of Predictive Clustering Trees			•	•	[45]	2007
	Ensembles independent of the multi-label classifier						
CBMLC	Clustering-Based for Multi-Label Classification		•		•	[47]	2009

used as clustering algorithm.

CBMLC obtains diverse classifiers by the selection of instances and also labels in each cluster.

3.5. Taxonomy of EMLCs

While overviewing state-of-the-art EMLCs we found some points by which to group or categorize them. As no taxonomy in the literature covers the characteristics of the EMLCs, we propose the following taxonomy for EMLCs, as shown in Table 1. First, the EMLCs can be categorized based on the multi-label method they are based, such as BR, LP, PCT or independent. Second, each EMLC can be categorized based on the way it generates diversity in the ensemble. Based on the taxonomy proposed in [49], we identified four ways or levels to generate diversity in EMLCs:

- Classifier level: at this level, different algorithms are used in the ensemble. The difference among classifiers can be given in several ways such as the use of different algorithms or the use of different parameters in the same algorithm. ECC and CDE are categorized at this level.
- Label level: at this level, each classifier of the ensemble is built over a different subset of labels. It usually implies a reduction in complexity of each classifier and also increase the diversity of classifiers in the ensemble. HOMER, AdaBoost.MH, RA*k*EL, TREMLC and CBMLC are categorized at this level.
- Feature level: at this level, each classifier of the ensemble uses a different subset of the features of the original training set, either disjoint or overlapping. This make each classifier focus on a subset of the input features, increasing the diversity and accuracy of the ensemble. MLS, TREMLC and RF-PCT are categorized at this level.
- Data level: at this level, each classifier is built over a different subset of the training dataset, either with or without replacement. It is a simple, effective and widely used method to generate a diverse ensemble [46]. EBR, ECC, HOMER, AdaBoost.MH, ELP, EPS, TREMLC, RF-PCT and CBMLC are categorized at this level.

These levels are not mutually exclusive, so each one of the EMLCs may appear in several groups simultaneously, i.e., an EMLC could be created by training each classifier of the ensemble over a subset of the labels and also over a subset of the features, belonging to both *label* and *feature levels*.

4. Experimental design

As mentioned in Section 1, the objective of this study is to perform an experimental comparison and analysis of the state-of-the-art EMLCs. In this section, first the chosen evaluation metrics and datasets are shown, then the default parameters of the EMLCs are presented and, finally, the experimental setup is explained.

4.1. Evaluation metrics

Evaluation metrics for multi-label classification are commonly distinguished in two groups: example-based metrics such as Hamming loss, accuracy or FMeasure_{ex} which are calculated for each instance, and label-based metrics such as FMeasure_{mac} and FMeasure_{mic} which are calculated with respect to labels. The formulation of the metrics used in this study are shown in Table 2, being Y the true labels, Z the predicted labels, m_{test} the number of instances of the test dataset and Δ computes the symmetric difference between two sets. In addition, *tp, fp* and *fn* refer to true positives, false positives and false negative of the contingency table respectively. A wider description of these evaluation metrics can be found in [9].

4.2. Datasets

Multi-label datasets have special characteristics that can be measured and identified by different characterization metrics. Density and diversity measure distribution of labels [50]. The density (*dens*) is defined as the mean number of relevant labels for each example divided by the total number of labels, and diversity (*div*) as the ratio of labelsets that appear in the dataset of the total of possible number of distinct labelsets. The *avgIR* measures the imbalance of the dataset by averaging the imbalance ratio of each label [51]. The greater the *avgIR* value, the greater the imbalance ratio of the labels of the dataset. Finally, the ratio of unconditionally dependent labels pairs by chi-square test (*rDep*)

Table 2
Multi-label evaluation metrics used in this study.

↓ Hamming loss	$\frac{1}{m_{test}}\sum_{i=1}^{m_{test}}\frac{1}{q}\lvert Z_i \Delta Y_i\rvert$
↑ Accuracy	$\frac{1}{m_{test}}\sum_{i=1}^{m_{test}}\frac{\lvert Z_i \cap Y_i\rvert}{\lvert Z_i \cup Y_i\rvert}$
↑ FMeasure_{ex}	$\frac{1}{m_{test}}\sum_{i=1}^{m_{test}}\frac{2\lvert Z_i \cap Y_i\rvert}{\lvert Z_i\rvert + \lvert Y_i\rvert}$
↑ FMeasure_{mac}	$\frac{1}{q}\sum_{i=1}^{q}\frac{2\cdot tp_i}{2\cdot tp_i + fn_i + fp_i}$
↑ FMeasure_{mic}	$\frac{\sum_{i=1}^{q} 2\cdot tp_i}{\sum_{i=1}^{q} 2\cdot tp_i + \sum_{i=1}^{q} fn_i + \sum_{i=1}^{q} fp_i}$

Table 3
Multi-label datasets.

	Domain	m	d	q	*dens*	*div*	*avgIR*	*rDep*	Ref
Flags	Image	194	19	7	0.485	0.422	2.255	0.381	[54]
CHD_49	Medicine	555	49	6	0.430	0.531	5.766	0.267	[55]
Water-quality	Chemistry	1060	16	14	0.362	0.778	1.767	0.473	[56]
Emotions	Music	593	72	6	0.311	0.422	1.478	0.933	[31]
3s_reuters1000	Text	294	1000	6	0.188	0.219	1.789	0.667	[57]
3s_guardian1000	Text	302	1000	6	0.188	0.219	1.773	0.667	[57]
3s_bbc1000	Text	352	1000	6	0.188	0.234	1.718	0.733	[57]
Birds	Audio	645	260	19	0.053	0.206	5.407	0.123	[58]
Yeast	Biology	2417	103	14	0.303	0.082	7.197	0.670	[59]
Scene	Image	2407	294	6	0.179	0.234	1.254	0.933	[17]
PlantPseAAC	Biology	978	440	12	0.090	0.033	6.690	0.318	[60]
HumanPseAAC	Biology	3106	440	14	0.085	0.027	15.289	0.418	[60]
Genbase	Biology	662	1186	27	0.046	0.048	37.315	0.157	[61]
Yelp	Text	10810	671	5	0.328	1.000	2.876	0.700	[62]
Medical	Text	978	1449	45	0.028	0.096	89.501	0.039	[63]
Slashdot	Text	3782	1079	22	0.054	0.041	19.462	0.273	[53]
Enron	Text	1702	1001	53	0.064	0.442	73.953	0.141	[37]
Langlog	Text	1460	1004	75	0.016	0.208	39.267	0.035	[53]
20NG	Text	19300	1006	20	0.051	0.003	1.007	0.984	[64]
Mediamill	Video	43910	120	101	0.043	0.149	256.405	0.342	[65]

measures the relationship among labels. The *rDep* is defined as the number of pairs of labels which are dependent at 99% confidence level by chi-square test divided by the total number of label pairs [52]. The greater *rDep*, the greater the relationship between the labels.

A set of 20 datasets from 8 different domains and with different characteristics is used in this study. The datasets range from 194 to 43910 instances, from 16 to 1449 features, and from 5 to 101 labels. Half of the datasets have on average less than 10% of the labels associated with an instance, but there are also several that have up to 40%. The diversity ranges from 0.003 (only the 0.3% of the possible labelsets appear in the dataset) to 1 (which means that all the possible labelsets appears in the dataset). The *avgIR* ranges from values near to 1 to values of more than 250, showing a great variety in the imbalance of the datasets. Finally, there is also a great variety in the degree of relationship among labels, with values of *rDep* ranging from near to zero (non related datasets) to near to 1 (highly related datasets). Table 3 shows the datasets including their domain and the values of the previously defined characterization metrics; the datasets are ordered by dimensionality, defined as $m \times d \times q$ according to [53].

All datasets were downloaded from a new repository of multi-label datasets[1]. Furthermore, the characterization of the datasets was performed using the MLDA tool [66].

4.3. Methods and configurations

All the methods described in Section 3, and listed in Table 4, were run using the default parameters proposed by their authors. For RA*k*EL and RA*k*EL++, two different configurations were used, according to the recommendations of their authors. Unless otherwise specified, all the methods used $n = 10$ classifiers in the ensemble, a threshold value of $t = 0.5$ and the C4.5 decision tree (Weka's J48 [67]) as a single-label base classifier. It has been shown that ensemble learning works well when decision trees are used as the base classifier [46]. Although some EMLCs use other base classifiers in addition to C4.5 in their original papers, it has been used as base classifier in almost all the studied EMLCs to obtain a greater consistency in the results.

4.4. Experimental setup

All the experiments were implemented using Meka [68] and Mulan [69] frameworks. Meka and Mulan are open-source Java frameworks for learning from multi-label datasets, which include a wide variety of state-of-the-art algorithms and provide an API to use their functionalities in Java code. In addition, CLUS library [19] was used to execute the RF-PCT algorithm. In order to ensure that all metrics are calculated in the same way, all the algorithms were executed by Meka. For that, the Meka's wrapper for Mulan algorithms and the Mulan's wrapper for CLUS algorithms have been used.

Table 4
Algorithms and default parameters proposed by their authors.

Algorithm	Parameters
EBR	$bagSizePercent = 100$
ECC	$bagSizePercent = 100$
MLS_{train}	–
MLS_{cv}	$numFolds = 10$
MLS_{ϕ}	$numFolds = 10$, $phiThreshold = 0.15$
HOMER	$clusteringAlgorithm = balancedk-means$, $c = 3$
AdaBoost.MH	$baseLearner =$ DecisionStump
ELP	$bagSizePercent = 100$
EPS	$bagSizePercent = 67$, $strategy = A$, $p = 1$, $b = 1$
RA*k*EL1	$k = q/2$, $n = 10$
RA*k*EL2	$k = 3$, $n = 2q$
RA*k*EL++1	$k = q/2$, $n = 10$
RA*k*EL++2	$k = 3$, $n = 2q$
RA*k*ELd	$k = 3$
TREMLC	$k = 3$, $n = 2q$, $bagSizePercent = 70$, $featurePercentage = 51$
CDE	$randomPartitions = 10000$
RF-PCT	$bagSizePercent = 100$
CBMLC	$clusteringAlgorithm = k-means$, $c = 5$

All the algorithms were executed over a stratified 5-folds cross-validation partitioning of the full dataset, using the Iterative Stratification method [70] to guarantee the distribution of labels in the partitions is as similar as possible. For algorithms which use random numbers (such as EBR, ECC, ELP, EPS, RA*k*EL, RA*k*EL++, RA*k*ELd, TREMLC, CDE, RF-PCT and CBMLC) 10 different seeds were used.

Since three versions of MLS and five versions of RA*k*EL are available, first comparisons among MLS and RA*k*EL methods are performed separately to determine which version of each algorithm is the best. Then, the best variants of MLS and RA*k*EL are used in the complete study.

Next, several experiments to compare different EMLCs based on their characteristics are performed. First, the performance of the EMLCs is evaluated given their imbalance ratio. Second, the EMLCs are

[1] http://www.uco.es/kdis/mllresources/.

evaluated taking into account the degree of dependency among labels. Then, the EMLCs are evaluated in terms of efficiency. Finally, an overall comparison of all EMLCs over all evaluation metrics is performed.

In order to compare the performance of different EMLCs, the Skillings-Mack's test [71,72] was performed for each metric. Skillings-Mack's test is similar to Friedman's test [73] but it can be used with missing values. The cases marked as DNF in the results are treated as missing values for the tests, except for training and testing times, where they are assigned the worst ranking value. In cases where the Skillings-Mack's test indicates that there exist significant differences in the performance of the algorithms, the Shaffer's post-hoc test [74] was used to perform multiple comparisons among all the methods. The use of Shaffer's test was proposed in [75], to the detriment of other tests, such as Nemenyi's [76]. Furthermore, the adjusted *p*-values [77] were considered in the analysis. The adjusted p-values provide more statistical information since they take into account the fact that multiple comparisons can be directly performed with any significance level. In this work, a significance level of $\alpha = 0.05$ was used.

5. Results and discussion

In this section the experimental results of the EMLCs are presented and discussed. First, results of the comparisons among variants of MLS and RA*k*EL are presented. Then, the results of each of the experiments are presented and discussed, including some tips how to select the best EMLC according to the characteristics of the dataset.

Datasets and detailed results for all experiments are fully described and publicly available to facilitate the replicability of the experiments and future comparisons at the KDIS Research Group website.[2] For the largest datasets, some algorithms did not finish the execution of a single fold within a day using the available resources.[3] These cases are marked as DNF (Did Not Finish) in the result tables.

5.1. Comparison among MLS and RAkEL variants

As previously mentioned, there are three variants of MLS and five variants of RA*k*EL. In order to simplify further study, first a comparison among those method is performed. The results in Table 5 show the Skillings-Mack test value, adjusted p-value and average ranking of each MLS variant and metric over all datasets. The best algorithm for each metric is marked in bold. Although there are no significant differences among MLS variants, MLS_{train} is the best algorithm in four of the five metrics.

Table 6 shows the results of the Skillings-Mack and Shaffer tests of each RA*k*EL variant. For each metric, the best algorithm is marked in bold and those algorithms which have significant differences with the best algorithm are marked with •. As seen, RA*k*EL2 is the best algorithm in four metrics, also being the only one that is not significantly different from the best algorithm in any case.

Based on these results, in further experiments only MLS_{train} and RA*k*EL2 are used.

5.2. Experiment 1: results depending on the imbalance of the datasets

In multi-label classification, one of the main problems is to deal with the imbalance of the data. Usually, some labels are very frequent while other are barely present in the dataset. This feature can have direct impact on performance of the algorithms, so we studied the performance of the EMLCs according to the imbalance of the dataset. Sorting the datasets by *avgIR*, we separate them into little, moderately, and very imbalanced datasets. Little imbalanced datasets are those with a *avgIR* < 2, moderately imbalanced are those with 2 ≤ *avgIR* < 20 and, any dataset with *avgIR* ≥ 20 is considered as very imbalanced.

Table 5
Skillings-Mack test results for the comparison among MLS variants.

Metric	Statistic	*p-value*	Rankings		
			MLS_{train}	MLS_{cv}	MLS_{ϕ}
Hamming loss	0.925	0.630	1.93	2.18	**1.90**
Accuracy	1.900	0.387	**1.75**	2.15	2.10
$FMeasure_{ex}$	1.875	0.392	**1.75**	2.13	2.13
$FMeasure_{mac}$	2.800	0.247	**1.80**	1.90	2.30
$FMeasure_{mic}$	1.575	0.455	**1.78**	2.08	2.15

$FMeasure_{mic}$ and $FMeasure_{mac}$ measure the performance of the algorithms over imbalanced data from two different points of view. While the former is biased by the frequency of occurrence of each label, the latter does not, giving equal importance to all labels independently of their frequency. $FMeasure_{mac}$ is more useful than $FMeasure_{mic}$ if infrequent or more imbalanced labels are present in evaluation of the classifier. Tables 7 and 8 show the datasets ordered by *avgIR* along with their results for $FMeasure_{mac}$ and $FMeasure_{mic}$, respectively. Further, these tables include the average rankings of each algorithm for little, moderately and very imbalanced datasets calculated as the Skillings-Mack's test.

As seen for $FMeasure_{mac}$, ELP is the best algorithm, on average, in little imbalanced datasets, CDE is the best on average for moderately imbalanced datasets and RA*k*EL2 is the best on average for very imbalanced datasets. The fact that both CDE and RA*k*EL2 split the output space into smaller labelsets for each base classifier causes that in cases of high degree of imbalance each base classifier has a more even distribution of labels than if all labels are taken into account at the same time, as in ELP. For very imbalanced datasets CDE is the best in the only two datasets where it finished, but due to its high complexity its average ranking deteriorates. The results for $FMeasure_{mic}$ are similar to those of $FMeasure_{mac}$ but in this case, for moderately imbalanced datasets ECC is the algorithm that performs better. Since $FMeasure_{mic}$ assigns more importance in the evaluation to more frequent labels if ECC predicts well these frequent labels; the performance according to this metric is higher. However it is common to try to predict correctly rare labels. Therefore, for a little imbalanced dataset, ELP is the best option, while for moderately and very imbalanced datasets RA*k*EL2 is the best one if all labels are considered to be equally important. If the high complexity of CDE does not matter, it also achieves good results for moderately and very imbalanced datasets.

5.3. Experiment 2: results depending on the relationship among labels

Another main challenge in multi-label classification is how to deal with the relationship among labels. The labels might be more or less correlated, and taking into account these correlations when learning a model could improve the performance. Sorting the datasets by *rDep*, we separated them into three groups. Those with *rDep* < 0.3 are considered as low dependent datasets, those with 0.3 ≤ *rDep* < 0.7 are considered as medium dependent datasets and those with *rDep* ≥ 0.7 are considered as highly dependent datasets.

$FMeasure_{ex}$ and Accuracy measure the multi-label prediction of each instance as a whole, which makes it useful for measuring the performance based on the relationship between labels. Tables 9 and 10 show the datasets ordered by *rDep* and their $FMeasure_{ex}$ and Accuracy results, respectively. The tables also include the average rankings for low, medium and highly dependent datasets.

For $FMeasure_{ex}$, ECC is the best algorithm on average, while for Accuracy RA*k*EL2 is the best in both low and medium dependent datasets. In both metrics ECC, RA*k*EL2 and CDE are the top three algorithms for low and medium datasets. These three methods take into

[2] http://www.uco.es/kdis/emlcreview/.
[3] The experiments have been performed on a machine with Debian 8, two Intel Core i7 CPUs at 2.67 GHz and 16GB of RAM.

Table 6
Skillings-Mack test results for the comparison among RA*k*EL variants.

Metric	Statistic	*p-value*	Rankings				
			RA*k*EL1	RA*k*EL2	RA*k*EL++1	RA*k*EL++2	RA*k*ELd
Hamming loss	37.510	1.41E−07	• 3.05	2.55	• 3.08	**1.68**	• 4.65
Accuracy	24.090	7.66E−05	2.30	**1.88**	• 3.73	• 3.30	• 3.80
$FMeasure_{ex}$	23.930	8.25E−05	2.20	**1.93**	• 3.75	• 3.58	• 3.55
$FMeasure_{mac}$	21.830	2.17E−04	2.38	**1.98**	• 3.85	• 3.75	3.05
$FMeasure_{mic}$	29.510	6.16E−06	2.30	**1.70**	• 3.83	• 3.33	• 3.85

Table 7
Results for $FMeasure_{mac}$ ↑ for all the EMLCs and datasets ordered by *avgIR*, including the average ranking ↓ of each algorithm for little, moderately and very imbalanced datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
20NG	0.650	0.671	0.622	0.609	0.000	**0.686**	**0.686**	0.656	0.574	DNF	0.399	0.328
Scene	0.706	**0.729**	0.647	0.586	0.000	0.704	0.703	0.701	0.700	0.720	0.711	0.598
Emotions	0.633	0.650	0.592	0.564	0.059	0.642	0.637	0.633	0.616	0.637	**0.653**	0.547
3s_bbc1000	0.083	0.123	0.158	0.230	0.000	0.204	0.174	0.195	0.150	0.200	0.051	**0.281**
Water-quality	0.501	0.525	0.465	0.520	0.082	0.466	0.278	0.523	0.508	0.508	**0.546**	0.484
3s_guardian1000	0.061	0.096	0.193	0.212	0.000	0.160	0.150	0.167	0.130	0.144	0.045	**0.294**
3s_reuters1000	0.076	0.104	0.160	0.196	0.000	0.170	0.148	0.185	0.131	0.164	0.058	**0.247**
	7.50	5.00	7.14	5.43	11.86	**4.36**	6.00	4.64	7.64	4.86	6.71	6.00
Flags	0.663	0.683	0.620	0.630	0.562	0.674	0.655	0.684	0.608	**0.689**	0.685	0.597
Yelp	0.710	0.721	0.683	0.675	0.000	0.706	0.706	**0.724**	0.647	**0.724**	0.614	0.600
Birds	0.230	0.239	0.234	**0.270**	0.000	0.250	0.207	0.251	0.191	0.260	0.230	0.179
CHD_49	0.497	**0.524**	0.464	0.492	0.270	0.511	0.511	0.517	0.505	0.516	0.520	0.505
PlantPseAAC	0.081	0.097	**0.160**	0.143	0.000	0.063	0.065	0.117	0.107	0.130	0.059	0.156
Yeast	0.387	0.401	0.395	0.403	0.122	0.380	0.375	0.409	0.389	**0.410**	0.396	0.396
HumanPseAAC	0.091	0.107	**0.150**	0.129	0.000	0.082	0.080	0.133	0.112	0.133	0.073	0.143
Slashdot	0.235	0.248	0.242	0.253	0.000	**0.301**	0.296	0.249	0.154	DNF	0.178	0.150
	7.31	4.50	6.00	5.13	11.88	6.25	7.38	3.13	8.19	**2.88**	7.13	7.50
Genbase	0.738	0.743	**0.747**	0.744	0.000	0.721	0.676	0.744	0.619	**0.747**	0.001	0.725
Langlog	0.032	0.039	0.051	**0.056**	0.000	0.017	0.025	0.045	0.031	DNF	0.005	0.037
Enron	0.153	0.158	0.152	**0.186**	0.013	0.121	0.116	0.164	0.131	DNF	0.111	0.104
Medical	0.352	0.360	0.371	0.343	0.000	0.338	0.327	**0.372**	0.274	**0.372**	0.026	0.178
Mediamill	0.187	0.179	0.211	0.175	0.009	DNF	0.164	**0.233**	0.033	DNF	0.200	0.074
	5.00	4.20	2.70	3.50	11.20	7.30	8.00	**2.20**	8.20	4.10	8.80	8.00

account the relationship among labels but in a softer way than other LP-based algorithms as ELP, so in cases where the dependency among labels is not very high, ECC and RA*k*EL2 are the best options. However, for highly dependent datasets ELP is the best algorithm on average. ELP considers all labels at a time, so the relationship among labels can be exploited more exhaustively than if labels are treated more independently.

5.4. Experiment 3: efficiency

Several EMLCs are computationally demanding, to the point that many of them did not build a single model within one day of computing. Thus, the efficiency of the EMLCs is a factor to be taken into account.

Sorting the datasets by dimensionality (defined as $m \times q \times d$), we separate them into three groups: small, medium and large datasets. We considere as small datasets those with *dimensionality* < 1E6, as medium datasets those with *dimensionality* ∈ [1E6, 1E8), and as large datasets those with *dimensionality* ≥ 1E8. In Tables 11 and 12 are shown the train and test times, respectively, for all the EMLCs and all the datasets. There are algorithms that are very fast but whose performance is bad, so not only the execution times but also the Hamming loss is shown in Table 13.

CBMLC generates several classifiers with a subset of similar instances and therefore possibly also with a subset of similar labels, which leads to less complex classifiers, being the fastest algorithm for small datasets and the second for medium and large datasets in both training and test times. However, CBMLC is one of the worst algorithms in terms of Hamming loss, regardless of the dimensionality of the dataset, being a fast algorithm but with a very low performance. Also for small datasets EPS is one of the fastest algorithms, getting also a good performance in terms of Hamming loss, so it is the best option for small datasets. On the other hand, for medium and large datasets, RF-PCT is the most efficient algorithm in both train and test. For larger datasets, RF-PCT, which reduces considerably the selection of the attributes in each node of the tree, has a lower complexity than other EMLCs and therefore higher efficiency. In terms of Hamming loss, RF-PCT results are not bad, so it is a great option for medium and large datasets if a very fast but not best in prediction algorithm is needed. Also for medium datasets EPS is one of the best algorithms in Hamming loss, being the best for large datasets. This fact, coupled with acceptable execution time, makes EPS the best option considering both execution time and performance, regardless of the dimensionality of the dataset. CDE did not finish with any of the large datasets, so it has not been assigned any average ranking value.

Table 8
Results for $FMeasure_{mic}$ ↑ for all the EMLCs and datasets ordered by *avgIR*, including the average ranking ↓ of each algorithm for little, moderately and very imbalanced datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
20NG	0.663	0.681	0.630	0.610	0.000	**0.692**	**0.692**	0.664	0.592	DNF	0.424	0.330
Scene	0.702	**0.722**	0.638	0.576	0.000	0.697	0.696	0.693	0.692	0.714	0.702	0.591
Emotions	0.653	0.666	0.599	0.572	0.105	0.660	0.654	0.648	0.628	0.652	**0.671**	0.557
3s_bbc1000	0.112	0.169	0.192	0.246	0.000	0.233	0.208	0.230	0.192	0.238	0.075	**0.294**
Water-quality	0.563	0.582	0.516	0.568	0.249	0.535	0.413	0.573	0.565	0.575	**0.595**	0.491
3s_guardian1000	0.094	0.144	0.240	0.226	0.000	0.213	0.203	0.211	0.179	0.181	0.076	**0.309**
3s_reuters1000	0.112	0.156	0.198	0.213	0.000	0.218	0.193	0.222	0.177	0.200	0.099	**0.264**
	7.21	5.00	6.93	6.00	11.86	**4.07**	5.93	4.86	7.64	4.57	6.79	6.29
Flags	0.753	0.760	0.720	0.736	0.711	0.749	0.745	0.761	0.745	**0.765**	0.771	0.651
Yelp	0.749	0.759	0.721	0.708	0.000	0.739	0.739	**0.754**	0.672	**0.759**	0.684	0.631
Birds	0.418	0.440	0.375	**0.391**	0.000	0.433	0.413	0.422	0.360	0.432	0.410	0.201
CHD_49	0.654	**0.677**	0.604	0.644	0.598	0.667	0.668	0.665	0.665	0.665	0.676	0.590
PlantPseAAC	0.161	0.205	**0.239**	0.203	0.000	0.148	0.148	0.218	0.200	0.220	0.160	0.204
Yeast	0.626	0.637	0.548	0.585	0.480	0.626	0.625	0.621	0.609	**0.631**	0.636	0.493
HumanPseAAC	0.246	0.292	**0.300**	0.270	0.000	0.243	0.240	0.316	0.266	0.312	0.248	0.206
Slashdot	0.464	0.476	0.473	0.457	0.000	**0.508**	0.513	0.480	0.349	DNF	0.394	0.179
	6.06	**2.56**	6.88	7.63	11.63	5.44	6.19	3.75	7.94	3.19	5.63	10.38
Genbase	0.989	0.988	**0.989**	0.987	0.000	0.979	0.977	0.989	0.903	**0.989**	0.000	0.978
Langlog	0.163	0.189	0.192	**0.150**	0.000	0.101	0.140	0.190	0.147	DNF	0.029	0.040
Enron	0.573	0.587	0.522	**0.526**	0.245	0.512	0.507	0.574	0.558	DNF	0.526	0.126
Medical	0.813	0.816	0.812	0.793	0.000	0.785	0.783	**0.813**	0.724	**0.815**	0.180	0.323
Mediamill	0.617	0.616	0.555	0.549	0.287	DNF	0.600	**0.618**	0.300	DNF	0.621	0.110
	3.20	2.80	4.30	5.90	10.70	8.30	7.60	**2.40**	7.40	4.40	7.80	9.60

Table 9
Results for $FMeasure_{ex}$ ↑ for all the EMLCs and datasets ordered by *rDep*, including the average ranking ↓ of each algorithm for low, medium and high dependent datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
Langlog	0.098	0.117	**0.130**	0.108	0.000	0.059	0.083	0.123	0.088	DNF	0.016	0.089
Medical	0.782	**0.795**	0.787	0.775	0.000	0.780	0.777	0.788	0.659	0.792	0.110	0.714
Birds	0.069	0.119	0.139	0.205	0.000	0.171	0.153	0.174	0.135	0.176	0.046	**0.283**
Enron	0.553	**0.581**	0.496	0.528	0.231	0.505	0.499	0.556	0.539	DNF	0.511	0.266
Genbase	0.172	0.222	**0.268**	0.224	0.000	0.170	0.168	0.254	0.201	0.245	0.174	0.259
CHD_49	0.623	**0.650**	0.565	0.615	0.580	0.640	0.642	0.640	0.638	0.639	0.650	0.561
Slashdot	0.363	0.391	0.375	0.392	0.000	0.447	**0.452**	0.378	0.251	DNF	0.289	0.430
	6.86	3.79	5.71	5.29	11.29	6.21	6.29	**3.50**	7.57	4.71	8.07	6.14
PlantPseAAC	0.522	**0.544**	0.513	0.506	0.000	0.514	0.514	0.539	0.455	0.532	0.662	0.499
Mediamill	**0.587**	**0.588**	0.529	0.527	0.297	DNF	0.574	0.589	0.282	DNF	**0.594**	0.319
Flags	0.722	0.734	0.685	0.713	0.660	0.721	0.717	0.734	0.728	0.741	**0.754**	0.633
HumanPseAAC	0.102	0.145	0.204	0.153	0.000	0.096	0.095	0.163	0.144	0.159	0.106	**0.240**
Water-quality	0.531	0.553	0.481	0.539	0.232	0.507	0.397	0.543	0.534	0.547	**0.568**	0.458
3s_guardian1000	0.597	**0.613**	0.524	0.559	0.456	0.600	0.599	0.599	0.586	0.606	0.616	0.496
3s_reuters1000	0.170	**0.182**	0.159	0.182	0.000	0.170	0.166	0.179	0.147	0.184	0.160	0.158
Yeast	0.057	0.098	0.187	0.190	0.000	0.154	0.145	0.156	0.126	0.124	0.047	**0.291**
	6.69	3.75	7.13	6.06	11.50	6.44	7.50	**3.63**	8.13	3.81	4.00	7.88
Yelp	0.991	0.990	**0.991**	0.988	0.000	0.986	0.985	0.991	0.830	0.991	0.000	0.986
3s_bbc1000	0.616	**0.675**	0.595	0.524	0.000	0.650	0.649	0.646	0.637	0.663	0.655	0.616
Scene	0.068	0.107	0.154	0.169	0.000	0.159	0.140	0.172	0.122	0.135	0.060	**0.226**
Emotions	0.594	0.621	0.554	0.523	0.061	0.615	0.609	0.608	0.583	0.606	**0.634**	0.539
20NG	0.540	0.580	0.521	0.547	0.000	**0.627**	**0.627**	0.556	0.451	DNF	0.277	0.587
	7.00	4.20	6.90	7.40	11.70	**4.00**	5.10	4.10	8.40	4.70	7.30	6.00

5.5. Experiment 4: general results

Once the results of the EMLCs have been studied depending on the characteristics of the datasets, we also studied them in terms of overall performance, taking into account all the evaluation metrics.

The statistic values and adjusted *p*-values obtained from the Skillings-Mack's test are shown in Table 14. As Skillings-Mack's test rejects the null hypothesis for all metrics at 95% confidence level, the Shaffer's post-hoc test was also performed. Significant differences among EMLC methods for all performance metrics at 95% confidence level are shown for example-based metrics in Fig. 1, for label-based metrics in Fig. 2, and for efficiency metrics in Fig. 3.

Table 10
Results for Accuracy ↑ for all the EMLCs and datasets ordered by *rDep*, including the average ranking ↓ of each algorithm for low, medium and high dependent datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
Langlog	0.232	0.249	**0.256**	0.233	0.142	0.198	0.220	0.252	0.222	DNF	0.157	0.219
Medical	0.752	**0.765**	0.756	0.745	0.000	0.756	0.753	0.757	0.627	0.761	0.101	0.692
Birds	0.065	0.113	0.126	0.175	0.000	0.162	0.146	0.160	0.126	0.165	0.045	**0.245**
Enron	0.442	**0.471**	0.390	0.411	0.150	0.399	0.397	0.443	0.429	DNF	0.405	0.209
Genbase	0.163	0.209	**0.234**	0.189	0.000	0.160	0.158	0.231	0.185	0.224	0.162	0.228
CHD_49	0.513	**0.540**	0.457	0.495	0.464	0.529	0.530	0.527	0.523	0.529	0.535	0.440
Slashdot	0.348	0.375	0.359	0.370	0.000	0.431	**0.436**	0.362	0.243	DNF	0.281	0.404
	7.00	**3.43**	5.86	5.57	11.29	5.86	6.00	3.71	7.36	4.50	8.29	6.57
PlantPseAAC	0.725	**0.739**	0.694	0.684	0.250	0.723	0.723	0.737	0.636	0.737	0.577	0.659
Mediamill	**0.489**	**0.489**	0.424	0.419	0.192	DNF	0.476	0.486	0.183	DNF	**0.489**	0.267
Flags	0.615	0.631	0.575	0.598	0.541	0.617	0.615	0.633	0.617	0.637	**0.644**	0.521
HumanPseAAC	0.099	0.142	0.180	0.129	0.000	0.094	0.094	0.153	0.136	0.150	0.104	**0.221**
Water-quality	0.392	0.411	0.347	0.394	0.151	0.369	0.281	0.401	0.392	0.405	**0.424**	0.320
3s_guardian1000	0.486	**0.506**	0.407	0.439	0.335	0.493	0.492	0.482	0.469	0.495	0.505	0.386
3s_reuters1000	0.596	**0.603**	0.544	0.555	0.456	0.594	0.589	0.586	0.563	0.591	0.579	0.501
Yeast	0.054	0.091	0.168	0.168	0.000	0.144	0.137	0.142	0.116	0.111	0.045	**0.248**
	5.88	**3.13**	7.06	7.06	11.50	5.75	6.94	4.31	7.63	4.00	5.38	7.88
Yelp	0.987	0.986	**0.988**	0.984	0.000	0.982	0.980	0.987	0.822	0.987	0.000	0.982
3s_bbc1000	0.600	**0.660**	0.562	0.489	0.000	0.636	0.636	0.624	0.617	0.641	0.639	0.599
Scene	0.062	0.098	0.139	0.144	0.000	0.149	0.132	0.157	0.112	0.120	0.057	**0.189**
Emotions	0.513	0.539	0.461	0.426	0.045	0.536	0.529	0.520	0.499	0.522	**0.548**	0.455
20NG	0.529	0.570	0.506	0.522	0.000	**0.623**	**0.623**	0.541	0.443	DNF	0.276	0.579
	6.80	4.20	6.60	7.80	11.70	**3.90**	5.00	4.40	8.40	4.60	7.30	6.10

Regarding the example-based metrics (Fig. 1) we see that EBR performed best for Hamming loss but it is the 6th for accuracy and the 9th for $FMeasure_{ex}$. ECC had best average ranking for accuracy and second for both Hamming loss and $FMeasure_{ex}$. Finally, RA*k*EL2 is the best for $FMeasure_{ex}$, second for accuracy but the 7th for Hamming loss, having significant differences with EBR at 95% confidence level.

For the label-based metrics (Fig. 2), RA*k*EL2 is the best for $FMeasure_{mac}$ and the second for $FMeasure_{mic}$, while ECC is the best in $FMeasure_{mic}$ and third for $FMeasure_{mac}$. That means that when all labels are considered to be equal, RA*k*EL2 performs better, but if the evaluation is biased by the frequency of the labels, ECC is the best choice. Anyhow, these two methods perform better on imbalanced problems

Table 11
Results for training time ↓ for all the EMLCs and datasets ordered by dimensionality, including the average ranking ↓ of each algorithm for small, medium and large datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
Flags	5.20E-01	5.27E-01	5.57E-01	4.47E-01	3.70E-01	**2.92E-01**	4.10E-01	5.57E-01	5.16E-01	2.62E+00	6.79E-01	4.01E-01
CHD_49	1.10E+00	1.07E+00	1.31E+00	8.25E-01	6.94E-01	7.02E-01	7.38E-01	1.06E+00	8.75E-01	9.61E+00	8.64E-01	**6.39E-01**
Water-quality	2.19E+00	2.29E+00	2.07E+00	1.36E+00	1.18E+00	1.28E+00	7.97E-01	1.74E+00	1.38E+00	4.93E+01	1.67E+00	**7.25E-01**
Emotions	1.54E+00	1.46E+00	1.60E+00	9.50E-01	1.00E+00	1.11E+00	9.46E-01	1.40E+00	1.18E+00	2.06E+01	1.03E+00	**7.43E-01**
	9.25	9.25	10.13	4.50	2.75	3.50	3.00	8.38	6.50	12.00	7.25	**1.50**
3s_reuters1000	6.97E+00	7.07E+00	4.68E+00	3.39E+00	4.04E+00	7.67E+00	7.35E+00	1.01E+01	3.38E+00	1.04E+02	**1.09E+00**	3.41E+00
3s_guardian1000	7.18E+00	7.32E+00	4.72E+00	3.31E+00	4.14E+00	7.87E+00	7.60E+00	1.06E+01	3.63E+00	1.06E+02	**1.12E+00**	3.37E+00
3s_bbc1000	9.52E+00	8.58E+00	6.08E+00	4.09E+00	4.39E+00	1.07E+01	9.82E+00	1.35E+01	4.27E+00	2.06E+02	**1.17E+00**	4.53E+00
Birds	5.67E+00	5.90E+00	4.68E+00	1.84E+00	3.60E+00	2.20E+00	1.63E+00	7.67E+00	3.83E+00	4.15E+02	1.77E+00	**1.27E+00**
Yeast	2.85E+01	1.85E+01	1.46E+01	4.19E+00	5.06E+00	9.49E+00	7.39E+00	1.99E+01	8.62E+00	6.96E+02	5.19E+00	**2.74E+00**
Scene	1.96E+01	1.85E+01	1.52E+01	4.53E+00	6.61E+00	1.00E+01	9.31E+00	1.37E+01	7.01E+00	3.14E+02	3.46E+00	**3.39E+00**
PlantPseAAC	3.89E+01	4.26E+01	3.24E+01	7.27E+00	2.07E+01	1.54E+01	1.67E+01	3.40E+01	1.44E+01	9.24E+02	**2.16E+00**	4.82E+00
HumanPseAAC	3.64E+02	3.94E+02	1.80E+02	4.70E+01	1.12E+02	5.53E+01	8.72E+01	2.57E+02	7.41E+01	6.59E+03	**5.71E+00**	2.31E+01
Genbase	9.25E+00	5.05E+00	4.69E+00	3.14E+00	8.05E+00	2.16E+00	1.94E+00	6.12E+00	3.39E+00	1.84E+03	**1.61E+00**	3.09E+00
Yelp	9.02E+02	7.78E+02	4.46E+02	1.78E+02	5.48E+01	2.53E+02	2.53E+02	4.93E+02	1.34E+02	8.57E+03	**7.15E+00**	4.84E+01
Medical	2.79E+01	3.45E+01	4.11E+01	8.02E+00	3.92E+01	1.60E+01	7.62E+00	3.32E+01	2.51E+01	2.88E+03	**2.34E+00**	2.75E+00
Slashdot	1.66E+03	1.77E+03	1.15E+03	1.11E+02	3.22E+02	8.14E+02	1.12E+03	1.13E+03	2.26E+02	DNF	**7.33E+00**	2.62E+02
Enron	7.99E+02	1.07E+03	6.07E+02	1.03E+02	4.89E+02	1.11E+02	1.19E+02	8.36E+02	2.25E+02	DNF	**4.67E+00**	1.83E+01
	9.31	9.46	7.85	3.15	5.85	6.35	5.81	9.54	4.69	12.00	**1.46**	2.54
Langlog	5.54E+02	7.56E+02	4.13E+02	5.21E+01	6.97E+02	6.05E+01	6.45E+01	5.48E+02	1.41E+02	DNF	**4.86E+00**	1.62E+01
20NG	2.13E+04	2.67E+04	1.22E+04	1.94E+03	2.16E+03	3.39E+03	3.33E+03	1.42E+04	3.49E+03	DNF	**2.80E+01**	5.64E+02
Mediamill	9.21E+03	1.38E+04	9.47E+03	**4.03E+02**	1.02E+03	DNF	2.61E+03	5.41E+03	1.27E+03	DNF	5.95E+02	4.67E+02
	9.00	10.67	8.00	2.33	6.00	7.17	5.33	8.00	6.00	11.83	**1.67**	2.00

Table 12
Results for test time ↓ for all the EMLCs and datasets ordered by dimensionality, including the average ranking ↓ of each algorithm for small, medium and large datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
Flags	1.85E-01	2.24E-01	1.60E-01	1.08E-01	8.74E-02	**8.30E-02**	1.24E-01	1.69E-01	2.40E-01	1.43E+00	2.91E-01	9.98E-02
CHD_49	4.85E-01	5.86E-01	5.26E-01	2.62E-01	2.90E-01	2.70E-01	2.83E-01	3.75E-01	5.59E-01	6.78E+00	3.36E-01	**2.09E-01**
Water-quality	1.66E+00	1.89E+00	1.16E+00	5.31E-01	3.87E-01	9.13E-01	3.94E-01	1.03E+00	1.60E+00	4.55E+01	6.66E-01	**3.60E-01**
Emotions	9.98E-01	1.02E+00	7.56E-01	3.43E-01	4.51E-01	5.61E-01	4.40E-01	7.67E-01	7.23E-01	1.69E+01	3.33E-01	**3.04E-01**
	9.00	10.50	7.75	3.25	3.50	4.00	4.00	7.50	9.00	12.00	6.00	**1.50**
3s_reuters1000	5.70E+00	5.70E+00	3.76E+00	1.60E+00	3.23E+00	5.57E+00	5.64E+00	8.21E+00	2.72E+00	8.79E+01	**2.81E-01**	2.03E+00
3s_guardian1000	5.87E+00	5.85E+00	3.46E+00	1.65E+00	3.32E+00	5.88E+00	5.56E+00	8.19E+00	2.95E+00	9.29E+01	**2.86E-01**	2.22E+00
3s_bbc1000	7.98E+00	7.26E+00	4.85E+00	2.29E+00	3.87E+00	8.62E+00	7.57E+00	1.13E+01	3.60E+00	2.61E+02	**2.83E-01**	2.92E+00
Birds	5.16E+00	5.41E+00	3.76E+00	1.00E+00	2.28E+00	1.71E+00	1.09E+00	6.92E+00	3.97E+00	3.98E+02	6.91E-01	**6.37E-01**
Yeast	2.77E+01	1.78E+01	1.35E+01	3.17E+00	3.67E+00	8.63E+00	6.77E+00	1.83E+01	1.36E+01	6.55E+02	**1.01E+00**	1.95E+00
Scene	1.83E+01	1.77E+01	1.34E+01	3.15E+00	5.11E+00	8.90E+00	8.43E+00	1.24E+01	8.77E+00	2.76E+02	**5.35E-01**	2.14E+00
PlantPseAAC	3.84E+01	4.18E+01	3.13E+01	6.25E+00	1.92E+01	1.48E+01	1.54E+01	3.22E+01	1.50E+01	8.58E+02	**5.86E-01**	3.78E+00
HumanPseAAC	3.64E+02	3.91E+02	1.78E+02	4.37E+01	1.11E+02	5.46E+01	8.48E+01	2.55E+02	8.71E+01	6.24E+03	**1.20E+00**	2.05E+01
Genbase	9.09E+00	4.91E+00	3.95E+00	2.02E+00	6.00E+00	1.47E+00	1.31E+00	4.58E+00	6.82E+00	1.83E+03	**1.11E+00**	1.58E+00
Yelp	9.02E+02	7.75E+02	4.46E+02	1.78E+02	5.38E+01	2.55E+02	2.54E+02	5.02E+02	2.60E+02	7.08E+03	**1.39E+00**	4.66E+01
Medical	2.73E+01	3.34E+01	4.11E+01	6.29E+00	4.29E+01	1.51E+01	7.23E+00	3.20E+01	3.88E+01	2.80E+03	2.91E+00	**1.92E+00**
Slashdot	1.66E+03	1.78E+03	1.10E+03	1.09E+02	3.19E+02	8.10E+02	1.12E+03	1.13E+03	2.67E+02	DNF	**2.75E+00**	2.59E+02
Enron	8.00E+02	1.08E+03	6.00E+02	1.01E+02	4.85E+02	1.12E+02	1.18E+02	8.35E+02	2.48E+02	DNF	**5.62E+00**	1.72E+01
	9.65	9.50	7.38	2.92	6.00	5.92	5.54	9.38	6.23	12.00	**1.15**	2.31
Langlog	5.57E+02	7.63E+02	4.14E+02	5.09E+01	7.01E+02	6.01E+01	6.38E+01	5.49E+02	1.68E+02	DNF	**8.13E+00**	1.51E+01
20NG	2.13E+04	2.67E+04	1.22E+04	1.96E+03	2.19E+03	3.39E+03	3.34E+03	1.42E+04	7.25E+03	DNF	**7.97E+00**	6.01E+02
Mediamill	9.47E+03	1.41E+04	9.38E+03	3.86E+02	1.00E+03	DNF	2.68E+03	5.44E+03	1.28E+03	DNF	**3.09E+02**	4.86E+02
	9.33	10.67	7.67	2.67	6.00	7.17	5.33	8.00	6.00	11.83	**1.00**	2.33

than other methods which make the problem even more imbalanced, such as ELP. Also CDE achieves good results in both metrics being the second for $FMeasure_{mac}$ and the third for $FMeasure_{mic}$. It is noted that CDE even though there are cases where it does not finish the execution, obtains competitive results.

AdaBoost.MH is the algorithm with the worst performance in four of the five metrics. This is given because despite combining several Decision Stump classifiers will improve the use of a single of these classifiers, it does not achieve good results against other EMLCs using a more powerful base classifier such as C4.5. CBMLC and TREMLC are

Table 13
Results for Hamming loss ↓ for all the EMLCs and datasets ordered by dimensionality, including the average ranking ↓ of each algorithm for small, medium and large datasets.

	EBR	ECC	MLS_{train}	HOMER	AdaB.MH	ELP	EPS	RA*k*EL2	TREMLC	CDE	RF-PCT	CBMLC
Flags	0.244	0.244	0.265	0.273	0.277	0.253	0.253	0.243	0.253	**0.237**	0.241	0.393
CHD_49	0.301	**0.295**	0.336	0.325	0.307	0.303	0.302	0.305	0.304	0.306	0.312	0.418
Water-quality	**0.292**	0.298	0.333	0.343	0.340	0.319	0.306	0.311	0.315	0.295	0.315	0.572
Emotions	**0.202**	0.204	0.249	0.269	0.302	0.208	0.210	0.221	0.226	0.214	0.209	0.315
	2.13	2.63	9.50	10.25	10.00	5.50	4.75	5.25	6.63	4.00	5.38	12.00
3s_reuters1000	0.210	0.222	0.251	0.301	**0.188**	0.219	0.218	0.238	0.225	0.219	0.201	0.405
3s_guardian1000	0.205	0.219	0.244	0.295	**0.188**	0.219	0.216	0.232	0.230	0.225	0.202	0.385
3s_bbc1000	0.202	0.215	0.246	0.293	**0.188**	0.213	0.214	0.227	0.217	0.215	0.199	0.380
Birds	**0.043**	**0.043**	0.055	0.061	0.053	0.044	0.044	0.049	0.049	0.047	0.045	0.277
Yeast	**0.206**	0.211	0.284	0.259	0.232	0.214	0.213	0.226	0.228	0.216	0.220	0.393
Scene	0.093	**0.092**	0.130	0.151	0.179	0.100	0.100	0.104	0.102	0.097	0.099	0.157
PlantPseAAC	0.093	0.098	0.137	0.143	**0.090**	0.094	0.094	0.109	0.107	0.105	0.097	0.270
HumanPseAAC	**0.085**	0.088	0.119	0.126	**0.085**	0.087	0.087	0.097	0.094	0.095	0.090	0.290
Genbase	**0.001**	**0.001**	**0.001**	**0.001**	0.046	0.002	0.002	**0.001**	0.008	**0.001**	0.046	0.002
Yelp	0.085	0.084	0.098	0.105	0.182	0.088	0.088	0.086	0.108	**0.082**	0.163	0.148
Medical	**0.010**	**0.010**	**0.010**	0.011	0.028	0.011	0.012	**0.010**	0.014	**0.010**	0.026	0.089
Slashdot	**0.042**	0.043	0.043	0.049	0.054	0.043	**0.042**	**0.042**	0.046	DNF	0.043	0.319
Enron	**0.047**	0.048	0.053	0.061	0.062	0.052	0.051	0.048	0.048	DNF	0.050	0.644
	2.27	3.96	8.23	9.38	6.92	5.19	4.73	6.38	7.73	6.12	5.77	11.31
Langlog	0.016	0.016	0.019	0.026	0.016	0.016	**0.015**	0.018	0.016	DNF	0.016	0.463
20NG	**0.029**	**0.029**	0.033	0.039	0.051	**0.029**	**0.029**	0.030	0.033	DNF	0.039	0.144
Mediamill	**0.027**	0.028	0.037	0.038	0.038	DNF	0.029	0.029	0.065	DNF	0.029	0.477
	2.67	3.00	7.17	8.67	7.33	4.17	**2.50**	5.67	6.67	–	5.67	10.67

Table 14
Skillings-Mack's test statistic values and *p*-values for all the evaluation metrics. The null hypothesis is rejected for all the metrics.

	Statistic	*p-value*
Hamming loss	118.78	.000000
Accuracy	78.50	.000000
FMeasure_{ex}	77.99	.000000
FMeasure_{mac}	91.36	.000000
FMeasure_{mic}	93.00	.000000
Training time	162.26	.000000
Test time	170.37	.000000

also two algorithms that usually are in the last positions of the rankings. Both algorithms reduce significantly the data in each classifier (CBMLC split the data into five different groups and TREMLC uses 70% of the instances and 51% of the features in each classifier), creating weaker models that finally do not achieve good results.

In order to evaluate the efficiency of the algorithms, those that did not finish the execution were assigned a considerably high execution time so that they get the worst ranking in those cases. In terms of efficiency (Fig. 3), CBMLC was the fastest algorithm in training, including significant differences with algorithms such as RA*k*EL2, EBR and ECC. The high efficiency of CBMLC is due to the decomposition of the data into several groups of similar data, building models over a dataset with a reduced number of instances and also possibly over a reduced set of labels, which leads to much simpler base classifiers. However, despite its speed, CBMLC does not have a good performance as previously demonstrated. On the other hand, RF-PCT was the algorithm with a shorter average test time since it builds the trees making decisions in small random subsets of features, including significant differences with RA*k*EL2, EBR and ECC among others. In training and testing times, CDE was the slowest algorithm, to the point that it did not finish running on five datasets.

Finally, average rankings for each EMLC and for all performance metrics (not including training and testing times) are shown in Table 15. In order to calculate the average ranking for each algorithm, the ranking values for each metric and all datasets have been averaged. Finally, a meta-ranking was also calculated for each algorithm as the average value of the rankings of all metrics, obtaining a unique value for each algorithm. As shown, ECC achieves the best performance overall for all metrics and datasets, followed by RA*k*EL2 and CDE. It was also shown that ECC is always among the top three algorithms for all evaluation metrics (not considering times), which further strengthens its overall good performance. However, it is one of the worst in both training and test times. On the other hand, CDE, despite being the third best algorithm, has an extremely high complexity which causes sometimes not finished running.

5.6. Guidelines to choose the best EMLC based on the characteristics of the data

In this section we summarize the tips and guidelines presented in the discussion of the different experiments to choose the EMLC that best fits to the dataset.

With respect to the imbalance level of the dataset, the results showed that ELP, which deal with all labels at once is the algorithm with best average performance for datasets with small imbalance (*avgIR* < 2), while RA*k*EL2, which considers the labels in small subsets obtaining a much more balanced output space, is the best algorithm on average for moderately and very imbalanced datasets (*avgIR* ≥ 2).

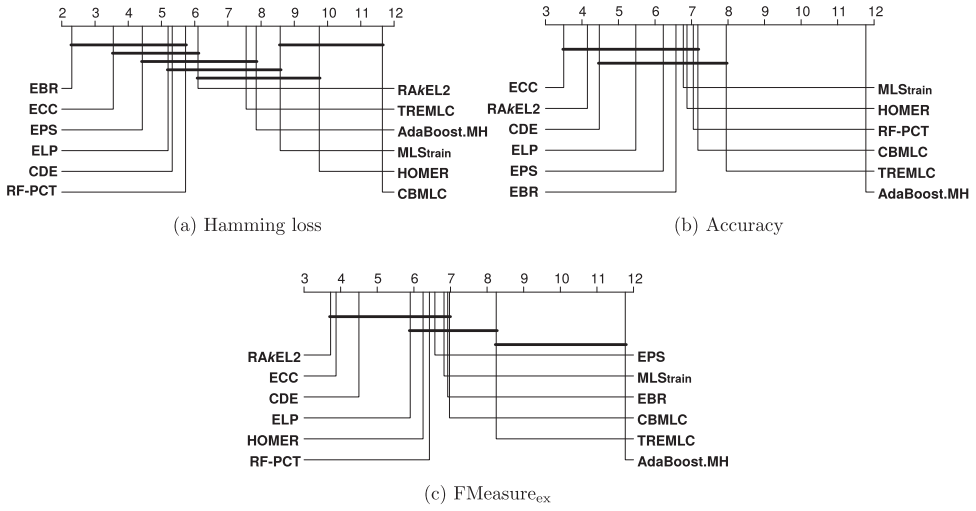
Regarding the relationship among labels, the results indicate that both RA*k*EL2 and ECC, which take into account the relationships between labels to a lesser extent, are good options for low and medium dependent datasets (*rDep* < 0.7). However, for highly dependent datasets (*rDep* ≥ 0.7) ELP is the best algorithm. Since ELP considers all possible relationships among labels, it can exploit the relationship among labels more exhaustively and therefore it achieves a better performance in highly dependent datasets.

In terms of efficiency, CBMLC was the fastest algorithm in training and test times for small datasets (*complexity* $<$ 1E6), however, it does not achieve a good performance. On the other hand, RF-PCT was the fastest in both training and test times for medium and large datasets (*complexity* $\geq$ 1E6), also getting an acceptable performance. It is worth mentioning EPS, a combination of good performance and fast algorithm, which is a good option if a fast but also accurate classifier is needed.

Finally, the results of the experimental comparison with all metrics, showed that ECC, followed by RA*k*EL, was the algorithm with best overall performance for all the metrics used.

6. Conclusions

In this paper, we presented an experimental review of the state-of-the-art EMLC methods, comparing a total of 18 methods over 20 datasets from different domains and with different characteristics. As a result of the study and the fact that no taxonomy for EMLCs was available in the literature, a novel categorization of the EMLC methods was also proposed. EMLC methods have been categorized based on which multi-label method they use, such as BR, LP, PCT or independent of the multi-label classifier. In addition, they have been categorized based on the way the ensemble is built, including the *classifier level,*

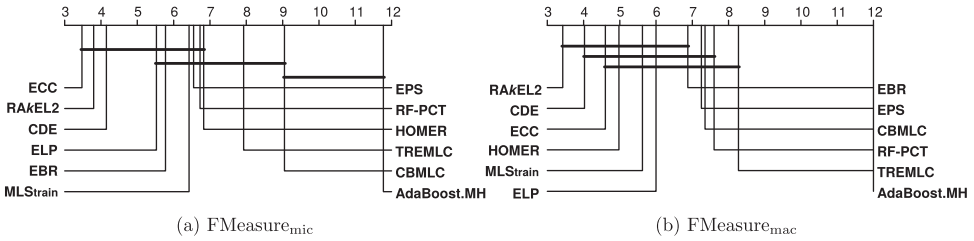


(a) Hamming loss

(b) Accuracy

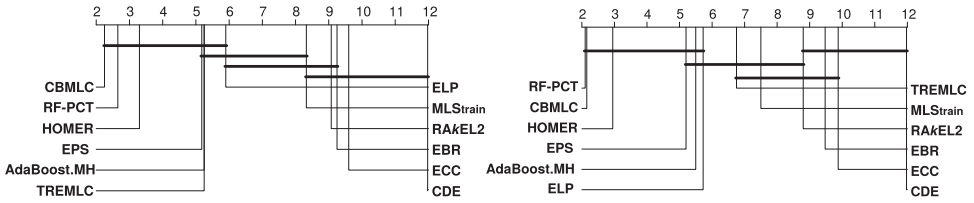
(c) FMeasure_{ex}

Fig. 1. Critical diagrams for the example-based metrics: results of the Shaffer's test at 95% confidence for (a) Hamming loss, (b) accuracy and (c) FMeasure_{ex}.



(a) $\text{FMeasure}_{\text{mic}}$ (b) $\text{FMeasure}_{\text{mac}}$

Fig. 2. Critical diagrams for the label-based metrics: results of the Shaffer's test at 95% confidence for (a) $\text{FMeasure}_{\text{mic}}$ and (b) $\text{FMeasure}_{\text{mac}}$.



(a) Training time (b) Test time

Fig. 3. Critical diagrams for the efficiency measures: results of the Shaffer's test at 95% confidence for (a) training time and (b) test time.

Table 15
Average rankings for all evaluation metrics and meta-ranking values for each EMLC.

	EBR	ECC	MLStrain	HOMER	AdaB.MH	ELP	EPS	RAkEL2	TREMLC	CDE	RF-PCT	CBMLC
Hamming loss	**2.30**	3.55	8.33	9.45	7.60	5.10	4.40	6.05	7.35	5.05	5.68	11.35
Accuracy	6.50	**3.50**	6.53	6.73	11.48	5.33	6.13	4.13	7.73	4.33	6.88	6.98
$\text{FMeasure}_{\text{ex}}$	6.83	3.88	6.58	6.13	11.48	5.75	6.48	**3.70**	8.00	4.35	6.25	6.80
$\text{FMeasure}_{\text{mac}}$	6.80	4.60	5.58	4.83	11.70	5.85	7.05	**3.43**	8.00	3.88	7.40	7.10
$\text{FMeasure}_{\text{mic}}$	5.75	**3.48**	6.25	6.63	11.48	5.38	6.45	3.80	7.70	3.98	6.58	8.75
Meta-rank	5.64	**3.80**	6.65	6.75	10.75	5.48	6.10	4.22	7.76	4.32	6.56	8.20

label level, feature level and *data level*.

The performance of the EMLCs was evaluated from different points of view and taking into account the characteristics of the datasets such as the imbalance, relationship, and dimensionality. Some guidelines were also given in order to choose the EMLC that best fits to the data in each case.

Acknowledgments

This research was supported by the Spanish Ministry of Economy and Competitiveness, Project TIN-2014-55252-P, and by FEDER funds. This research was also supported by the Spanish Ministry of Education under FPU Grant FPU15/02948.

References

[1] L. Rokach, Ensemble-based classifiers, Artif. Intell. Rev. 33 (1) (2010) 1–39, http://dx.doi.org/10.1007/s10462-009-9124-7.
[2] M. Wozniak, M. Graña, E. Corchado, A survey of multiple classifier systems as hybrid systems, Inf. Fusion 16 (2014) 3–17. Special Issue on Information Fusion in Hybrid Intelligent Fusion Systems. doi:10.1016/j.inffus.2013.04.006 .
[3] T.G. Dietterich, Ensemble Methods in Machine Learning, Springer Berlin Heidelberg, pp. 1–15.
[4] W. Leigh, R. Purvis, J.M. Ragusa, Forecasting the {NYSE} composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support, Decis. Supp. Syst. 32 (4) (2002) 361–377, http://dx.doi.org/10.1016/S0167-9236(01)00121-X.
[5] A.C. Tan, D. Gilbert, Y. Deville, Multi-class protein fold classification using a new ensemble machine learning approach, Genome Informat. 14 (2003) 206–217, http://dx.doi.org/10.11234/gi1990.14.206.
[6] P. Mangiameli, D. West, R. Rampal, Model selection for medical diagnosis decision support systems, Decis. Supp. Syst. 36 (3) (2004) 247–259, http://dx.doi.org/10.1016/S0167-9236(02)00143-4.
[7] H.-J. Lin, Y.-T. Kao, F.-W. Yang, P.S.P. Wang, Content-based image retrieval trained by adaboost for mobile application, Int. J. Pattern Recognit. Artif. Intell. 20 (04) (2006) 525–541, http://dx.doi.org/10.1142/S021800140600482X.
[8] A. Schclar, A. Tsikinovsky, L. Rokach, A. Meisels, L. Antwarg, Ensemble methods for improving the performance of neighborhood-based collaborative filtering, Proceedings of the Third ACM Conference on Recommender Systems, RecSys '09, (2009), pp. 261–264, http://dx.doi.org/10.1145/1639714.1639763.
[9] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv. 47 (3) (2015).
[10] F. Herrera, F. Charte, A. Rivera, M. del Jesus, Multilabel Classification. Problem analysis, metrics and techniques, Springer, 2016, http://dx.doi.org/10.1007/978-3-319-41111-8.
[11] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Deroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognit. 45 (9) (2012) 3084–3104.
[12] G. Nasierding, A. Kouzani, Empirical study of multi-label classification methods for image annotation and retrieval, (2010), pp. 617–622, http://dx.doi.org/10.1109/DICTA.2010.113.
[13] P. Brandt, D. Moodley, A.W. Pillay, C.J. Seebregts, T. de Oliveira, An Investigation of Classification Algorithms for Predicting HIV Drug Resistance without Genotype Resistance Testing, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 236–253. doi:10.1007/978-3-642-53956-5_16.
[14] N.-Y. Nair-Benrekia, P. Kuntz, F. Meyer, Learning from multi-label data with interactivity constraints: an extensive experimental study, Expert Syst. Appl. 42 (13) (2015) 5723–5736, http://dx.doi.org/10.1016/j.eswa.2015.03.006.
[15] E. Gibaja, S. Ventura, Multi-label learning: a review of the state of the art and ongoing research, WIREs Data Mining Knowl Discov 2014. doi:10.1002/widm.1139.
[16] G. Tsoumakas, I. Katakis, I. Vlahavas, Data Mining and Knowledge Discovery Handbook, Part 6, Springer, pp. 667–685.
[17] M. Boutell, J. Luo, X. Shen, C. Brown, Learning multi-label scene classification, Pattern Recognit. 37 (2004) 1757–1771.
[18] A. Clare, A. Clare, R.D. King, Knowledge discovery in multi-label phenotype data, Lecture Notes in Computer Science, Springer, 2001, pp. 42–53.
[19] H. Blockeel, L.D. Raedt, J. Ramon, Top-down induction of clustering trees, Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 55–63.
[20] M. Petrovskiy, Paired comparisons method for solving multi-label learning problem, Proceedings of the Sixth International Conference on Hybrid Intelligent Systems, HIS '06, (2006), p. 42, http://dx.doi.org/10.1109/HIS.2006.54.
[21] J. Li, J. Xu, A fast multi-label classification algorithm based on double label support vector machine, Proceedings of the 2009 International Conference on Computational Intelligence and Security (CIS 09), (2009), pp. 30–35.
[22] K. Crammer, Y. Singer, A family of additive online algorithms for category ranking, J. Mach. Learn. Res. 3 (2003) 1025–1058.
[23] M.-L. Zhang, Z.-H. Zhou, Multi-label neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (2006) 1338–1351.
[24] M.-L. Zhang, Z.-H. Zhou, A k-Nearest Neighbor Based Algorithm for Multi-label Classification, Proceedings of the IEEE International Conference on Granular

Computing (GrC), 2 The IEEE Computational Intelligence Society, Beijing, China, 2005, pp. 718–721.
[25] W. Cheng, E. Hullermeier, Combining instance-based learning and logistic regression for multilabel classification, Mach. Learn. 76 (2-3) (2009) 211–225.
[26] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (3) (2011) 335–359.
[27] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[28] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, ECML 09: 20th European conference on machine learning, (2009), p. 254269.
[29] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, I. Vlahavas, Correlation-based pruning of stacked binary relevance models for multi-label learning, 1st International Workshop on Learning from Multi-Label Data (MLD'09), (2009), pp. 101–116.
[30] J. Cohen, P. Cohen, S.G. West, L.S. Aiken, Applied Multiple Regression / Correlation Analysis for the Behavioral Sciences, (2002).
[31] G. Tsoumakas, I. Katakis, I. Vlahavas, Effective and efficient multilabel classification in domains with large number of labels, Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD08), (2008).
[32] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1) (1997) 119–139, http://dx.doi.org/10.1006/jcss.1997.1504.
[33] L. Breiman, Arcing classifiers, Ann. Stat. 26 (3) (1998) 801–824.
[34] Y. Freund, R.E. Schapire, et al., Experiments with a new boosting algorithm, icml, 96 (1996), pp. 148–156.
[35] R. Maclin, D. Opitz, An empirical evaluation of bagging and boosting, Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence, AAAI'97/IAAI'97, (1997), pp. 546–551.
[36] R.E. Schapire, Y. Singer, Boostexter: A boosting-based system for text categorization, Mach. Learn. 39 (2000) 135–168.
[37] J. Read, B. Pfahringer, G. Holmes, Multi-label classification using ensembles of pruned sets, Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 995–1000.
[38] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multi-label classification, IEEE Trans. Knowl. Data Eng. 23 (7) (2011) 1079–1089.
[39] L. Rokach, A. Schclar, E. Itach, Ensemble methods for multi-label classification, Expert Syst. Appl. 41 (16) (2014) 7507–7523, http://dx.doi.org/10.1016/j.eswa.2014.06.015.
[40] G. Nasierding, A.Z. Kouzani, G. Tsoumakas, A triple-random ensemble classification method for mining multi-label data, ICDMW 2010, The 10th IEEE International Conference on Data Mining Workshops, Sydney, Australia, 13 December 2010, (2010), pp. 49–56, http://dx.doi.org/10.1109/ICDMW.2010.139.
[41] L. Tenenboim, L. Rokach, B. Shapira, Multi-label classification by analyzing labels dependencies, Proceedings of the 1st international workshop on learning from multi-label data, Bled, Slovenia, (2009), pp. 117–132.
[42] P.E. Greenwood, M.S. Nikulin, A guide to chi-squared testing, Wiley-Interscience 280 (1996).
[43] L. Tenenboim-Chekina, L. Rokach, B. Shapira, Identification of label dependencies for multi-label classification, Working Notes of the Second International Workshop on Learning from Multi-Label Data, (2010), pp. 53–60.
[44] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Wadsworth and Brooks, 1984.
[45] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Ensembles of Multi-Objective Decision Trees, Springer Berlin Heidelberg, pp. 624–631. doi:10.1007/978-3-540-74958-5_61.
[46] L. Rokach, Decision forest: twenty years of research, Inf. Fusion 27 (2016) 111–125, http://dx.doi.org/10.1016/j.inffus.2015.06.005.
[47] G. Nasierding, G. Tsoumakas, A.Z. Kouzani, Clustering based multi-label classification for image annotation and retrieval, Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11-14 October 2009, (2009), pp. 4514–4519, http://dx.doi.org/10.1109/ICSMC.2009.5346902.
[48] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Inc., 1988.
[49] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience, 2004.
[50] G. Tsoumakas, I. Katakis, Multi-label classification: An overview, Int. J. Data Warehousing Mining 3 (3) (2007) 1–13.
[51] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: measures and random resampling algorithms, Neurocomputing 163 (Supplement C) (2015) 3–16, http://dx.doi.org/10.1016/j.neucom.2014.08.091.
[52] L. Chekina, L. Rokach, B. Shapira, Meta-learning for selecting a multi-label classification algorithm, (2011), pp. 220–227.
[53] J. Read, Scalable multi-label classification, Ph.D. Thesis, University of Waikato (2010).
[54] E. Goncalves, A. Plastino, A.A. Freitas, A genetic algorithm for optimizing the label ordering in multi-label classifier chains, IEEE 25th International Conference on Tools with Artificial Intelligence, IEEE Computer Society Conference Publishing Services (CPS), 2013, pp. 469–476.
[55] H. Shao, G. Li, G. Liu, Y. Wang, Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine, Sci. Chin. Inform. Sci. 56 (5) (2013) 1–13, http://dx.doi.org/10.1007/s11432-011-4406-5.
[56] H. Blockeel, S. Deroski, J. Grbovi, Simultaneous prediction of multiple chemical parameters of river water quality with tilde, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 1704 (1999), pp. 32–40.
[57] D. Greene, P. Cunningham, A matrix factorization approach for integrating multiple data views, Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part I, ECML PKDD '09, (2009), pp. 423–438, http://dx.doi.org/10.1007/978-3-642-04180-8_45.
[58] F. Briggs, Y. Huang, R. Raich, K. Eftaxias, Z. Lei, W. Cukierski, S.F. Hadley, A. Hadley, M. Betts, X.Z. Fern, J. Irvine, L. Neal, A. Thomas, G. Fodor, G. Tsoumakas, H.W. Ng, T.N.T. Nguyen, H. Huttunen, P. Ruusuvuori, T. Manninen, A. Diment, T. Virtanen, J. Marzat, J. Defretin, D. Callender, C. Hurlburt, K. Larrey, M. Milakov, The 9th annual MLSP competition: new methods for acoustic classification of multiple simultaneous bird species in a noisy environment, IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2013, Southampton, United Kingdom, September 22-25, 2013, (2013), pp. 1–8, http://dx.doi.org/10.1109/MLSP.2013.6661934.
[59] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS'01, (2001), pp. 681–687.
[60] J. Xu, J. Liu, J. Yin, C. Sun, A multi-label feature extraction algorithm via maximizing feature variance and feature-label dependence simultaneously, Knowl.-Based Syst. 98 (2016) 172–184, http://dx.doi.org/10.1016/j.knosys.2016.01.032.
[61] S. Diplaris, G. Tsoumakas, P. Mitkas, I. Vlahavas, Protein classification with multiple algorithms, Proc. 10th Panhellenic Conference on Informatics (PCI 2005), (2005), pp. 448–456.
[62] Yelp dataset challenge, (http://www.ics.uci.edu/~vpsaini/). Last access: 26-06-2017.
[63] J.P. Pestian, C. Brew, P. Matykiewicz, D.J. Hovermale, N. Johnson, K.B. Cohen, W. Duch, A shared task involving multi-label classification of clinical free text, Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing (BioNLP '07), (2007), pp. 97–104.
[64] The 20 newsgroups data set, (http://qwone.com/~jason/20Newsgroups/). Last access: 26-06-2017.
[65] C. Snoek, M. Worring, J. van Gemert, J.-M. Geusebroek, A. Smeulders, The challenge problem for automated detection of 101 semantic concepts in multimedia, Proceedings of ACM Multimedia, (2006), pp. 421–430.
[66] J.M. Moyano, E.L. Gibaja, S. Ventura, MLDA: a tool for analyzing multi-label datasets, Knowl.-Based Syst. 121 (2017) 1–3, http://dx.doi.org/10.1016/j.knosys.2017.01.018.
[67] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[68] Meka: a multi-label extension to weka, (http://meka.sourceforge.net/). Last access: 31-03-2017.
[69] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: a java library for multi-label learning, J. Mach. Learn. Res. 12 (2011) 2411–2414.
[70] K. Sechidis, G. Tsoumakas, I. Vlahavas, On the stratification of multi-label data, Lect. Notes Comput. Sci. 6913 LNAI (PART 3) (2011) 145–158.
[71] M. Chatfield, A. Mander, The skillingsmack test (friedman test when there are missing data), Stata J. 9 (2) (2009) 299–305.
[72] P. Srisuradetchai, Skillings.mack: the skillings-mack test statistic for block designs with missing observations, (https://CRAN.R-project.org/package=Skillings.Mack). Last access: 12-12-2017.
[73] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Ann. Math. Statist. 11 (1) (1940) 86–92, http://dx.doi.org/10.1214/aoms/1177731944.
[74] J.P. Shaffer, Modified sequentially rejective multiple test procedures, J. Am. Stat. Assoc. 81 (395) (1986) 826–831.
[75] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (Dec) (2008) 2677–2694.
[76] P. Nemenyi, Distribution-free multiple-comparisons, Pricenton University (USA), 1963 Ph.D. Thesis.
[77] S.P. Wright, Adjusted p-values for simultaneous inference, Biometrics (1992) 1005–1013.

Chapter 4

Evolutionary approach to evolve a whole EMLC: EME

In spite of the wide range of EMLCs already proposed in the literature, many of them do not consider some characteristics of the data when building the models. For instance, the performance of ECC directly relies on the chain ordering, but it selects the chains randomly, not considering any of the characteristics of the data. Similarly, RA*k*EL selects subsets of labels to build each member, but they are randomly created without considering for example the relationship among these labels. On the other hand, some of the EMLCs are still not able to deal with these special characteristics of the multi-labeled data, as EBR, which is not able to model the dependencies among labels.

Aiming to improve the predictive performance of state-of-the-art EMLCs, we propose an evolutionary approach for the automatic generation of EMLCs, called EME, where the characteristics of the data are taken into account in the building phase. The EMLC generated by EME is based on the same idea as RA*k*EL, i.e., each base classifier is focused on modeling a small subset of k labels. In this way, it is able to model the relationship among groups of labels, but drastically reducing the complexity of each of them in scenarios with a high dimensionality of the output space. However, while RA*k*EL selects those k-labelsets just randomly, and thus it does not ensure that all labels are being considered or the number of times that each of them appears in the ensemble, in EME the k-labelsets are selected by looking for an optimal structure of the ensemble, and also ensuring that all labels appear a similar

number of times.

In EME, each individual of the population in the EA represents an EMLC. These individuals are initially randomly created, but as the generations go by, they evolve towards more promising combinations of classifiers into the ensemble. Individuals are evaluated not only by considering their predictive performance, but also taking into account the number of times that each label appears in the ensemble. In this way, we look for EMLCs that are not only accurate but that also that do not neglect some labels regardless of their frequency or ability to be predicted, thus trying to improve their generalization ability.

On the other hand, we propose a mutation operator for the EA where the relationships among labels are considered. For this purpose, for a given classifier inside the ensemble, labels that are more related with labels already appearing in the classifier have more chance to mutate and therefore to be included in the classifier than those that are not related with them. However, all labels still have a small chance to mutate, but we bias the evolution to the achievement of EMLCs dealing with subsets of labels that are related among them.

So, both RA*k*EL and EME deal with all the main problems of MLC: I) they reduce the dimensionality of the output space in each base classifier, II) thus leading to a reduction of the label imbalance, and III) being able to model the label relationships in each base classifier. Nonetheless, while RA*k*EL does not consider neither the relationship among labels nor the imbalance in its building phase, EME does.

EME is based upon a generational elitist algorithm, so it ensures that the best individual in the last generation is also the best individual so far in the EA process. Therefore, as each individual represents an entire ensemble, the best individual is returned at the end of the evolution.

In the experimental study, we first conduct a preliminary study to select the best parameters for the evolutionary algorithm, involving four datasets that are then not used in the comparison with state-of-the-art methods (in order to not to bias the final results). Later, we compare EME not only to other standard MLC methods but also to state-of-the-art EMLCs. Compared to EMLCs, it is demonstrated that EME has a better and more consistent performance in overall, being the best method in

one metric and also being the only method that did not perform significantly worse than the best method in any case. Tables with full results are available online[1].

Furthermore, EME has a better overall performance than RA*k*EL in four out of five evaluation metrics, and it also performs significantly better than RA*k*EL in two of the metrics. That proves that the fact of evolving the *k*-labelsets towards more promising combinations of labels instead of just selecting them randomly lead EME to achieve a better predictive performance.

Following, we present the paper associated with this chapter of the thesis [J5].

[1] https://www.uco.es/kdis/eme/

Title	An evolutionary approach to build ensembles of multi-label classifiers
Authors	Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastián Ventura
Journal	Information Fusion
Volume	50
Pages	168 - 180
Year	2019
DOI	10.1016/j.inffus.2018.11.013

IF (JCR 2018)	10.716
Category	Computer Science - Artificial Intelligence
Position	3/133 (Q1)
Cites (27/05/2020)	2 (WoS), 2 (Scopus), 4 (Google Scholar)

Information Fusion 50 (2019) 168–180

Contents lists available at ScienceDirect

Information Fusion

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

An evolutionary approach to build ensembles of multi-label classifiers

Jose M. Moyano [a,e], Eva L. Gibaja [a,e], Krzysztof J. Cios [b,c], Sebastián Ventura [a,d,e,*]

[a] Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain
[b] Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284, USA
[c] Polish Academy of Sciences, Institute of Theoretical and Applied Informatics, Gliwice, Poland
[d] Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
[e] Knowledge Discovery and Intelligent Systems in Biomedicine Laboratory, Maimonides Biomedical Research Institute of Córdoba, Spain

ARTICLE INFO

Keywords:
Multi-label classification
Ensemble
Evolutionary algorithm

ABSTRACT

In recent years, the multi-label classification task has gained the attention of the scientific community given its ability to solve problems where each of the instances of the dataset may be associated with several class labels at the same time instead of just one. The main problems to deal with in multi-label classification are the imbalance, the relationships among the labels, and the high complexity of the output space. A large number of methods for multi-label classification has been proposed, but although they aimed to deal with one or many of these problems, most of them did not take into account these characteristics of the data in their building phase. In this paper we present an evolutionary algorithm for automatic generation of ensembles of multi-label classifiers by tackling the three previously mentioned problems, called Evolutionary Multi-label Ensemble (EME). Each multi-label classifier is focused on a small subset of the labels, still considering the relationships among them but avoiding the high complexity of the output space. Further, the algorithm automatically designs the ensemble evaluating both its predictive performance and the number of times that each label appears in the ensemble, so that in imbalanced datasets infrequent labels are not ignored. For this purpose, we also proposed a novel mutation operator that considers the relationship among labels, looking for individuals where the labels are more related. EME was compared to other state-of-the-art algorithms for multi-label classification over a set of fourteen multi-label datasets and using five evaluation measures. The experimental study was carried out in two parts, first comparing EME to classic multi-label classification methods, and second comparing EME to other ensemble-based methods in multi-label classification. EME performed significantly better than the rest of classic methods in three out of five evaluation measures. On the other hand, EME performed the best in one measure in the second experiment and it was the only one that did not perform significantly worse than the control algorithm in any measure. These results showed that EME achieved a better and more consistent performance than the rest of the state-of-the-art methods in MLC.

1. Introduction

In recent years, the Multi-Label Classification (MLC) task has gained the attention of the scientific community given its ability to solve problems where each of the instances may be associated to several class labels at the same time, instead of just one. Let be $\mathcal{L} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ the set of q different binary labels (with $q > 2$), and $\mathcal{X}$ the set of m instances, each composed by d input features; let us define the multi-label classification task as learning a mapping from an example $\boldsymbol{x}_i \in \mathcal{X}$ to a set of labels $\boldsymbol{y}_i \subseteq \mathcal{L}$. Labels in the set $\boldsymbol{y}_i$ are called relevant labels, and the rest ($\overline{\boldsymbol{y}_i}$) are called irrelevant. A great deal of real-world problems have been successfully solved thanks to the application of MLC, such as social networks mining, where each user could be subscribed to several groups of interest [1]; multimedia annotation, where each image or multimedia item could be associated to several class labels [2]; and text categorization, where each document could be categorized in several topics simultaneously [3]; among others.

The most challenging problems in MLC are dealing with the imbalance of the data [4], modeling compound dependencies among the labels [5], and the possible high dimensionality of the output space [6]. In many problems the labels do not appear with the same frequency in the dataset, with some labels appearing in most of the instances and other that are barely present, appearing in a few instances. This might lead to an imbalanced dataset where the frequent labels could be much better predicted than the infrequent ones, as there is very little information about the infrequent labels. Besides, labels are not usually independent but tend to be related to each other, where a label may appear more frequently with some labels than with others. The fact of modeling, or lack of, compound dependencies among labels has a decisive effect not only on the predictive performance of the model but also on its

* Corresponding author.
E-mail address: sventura@uco.es (S. Ventura).

https://doi.org/10.1016/j.inffus.2018.11.013
Received 12 April 2018; Received in revised form 8 November 2018; Accepted 25 November 2018
Available online 26 November 2018

complexity. The complexity of the model is also usually related to the size of the output space. The greater the number of labels, the higher the complexity of the model, which can make the problem intractable.

In order to try to overcome these problems, several methodologies have been proposed in the literature. For example, Pruned Sets (PS) [7] was proposed in order to reduce the imbalance in the final problem. Besides, to overcome the problem of modelling the compound dependencies among labels, Classifier Chains (CC) [5] considered the relationship among different binary methods that originally did not take into account. For the output dimensionality problem, RAndom *k*-labELsets (RA*k*EL) [8] divided the label space into smaller subsets, resulting in less complex output spaces. Furthermore, the continuous stream of input data is a growing problem in many data mining tasks, and it has been also successfully addressed in MLC [9,10]. Many of these proposed methods were based on the combination of several classifiers. However, in MLC only those methods that combine several classifiers which are able to deal with multi-label data are considered as Ensembles of Multi-Label Classifiers (EMLCs) [11]. On the other hand, besides tackling the aforementioned problems, ensembles usually perform better than single classifiers. One of the ways to obtain an ensemble that outperform each of the individuals classifiers is to combine a set of diverse classifiers [12,13]. Despite this fact, many of the proposed ensemble methods in the literature generate diversity only by random sampling of attributes, instances, or labels for each classifier, but not ensuring that the entire ensemble is diverse enough.

In this paper, we propose an evolutionary approach for the automatic generation of ensembles of diverse and competitive multi-label classifiers. The algorithm, called Evolutionary Multi-Label Ensemble (EME), takes into account characteristics of the multi-label data such as the relationships among the labels, imbalance of the data, and complexity of the output space. The ensemble is based on projections of the label space, considering in this way the relationships among the labels but also reducing the computational cost in cases where the output space is complex. These subsets of labels are not only randomly selected but also they evolve with the generations of the evolutionary algorithm, looking for the combinations that perform the best. Also, a novel mutation operator is proposed, so that it considers the relationship among labels favouring more related combinations of labels. Further, EME takes into account all the labels approximately the same number of times in the ensemble, regardless of their frequency or its ease to be predicted; so that the imbalance of the data is considered and the infrequent labels are not ignored. For that, the fitness function takes into account both the predictive performance of the model and the number of times that each label is considered in the ensemble. Finally, the diversity of the ensemble is not taken into account explicitly, but the ensembles evolve selecting their classifiers based on their overall performance.

The experimental study carried out over fourteen multi-label datasets compared EME with classic state-of-the-art methods in MLC and also other EMLCs using five evaluation measures. The first experiment determined that EME performed significantly better than classic MLC methods in three of the five evaluation measures. In the second experiment, EME achieved the best performance in only one measure, but it was the only algorithm that did not perform significantly worse than any of the control algorithm for any evaluation measures. These results showed that EME achieved a better and more consistent performance than the rest of the state-of-the-art methods in MLC.

The rest of the article is organized as follows: Section 2 includes related work in multi-label classification, Section 3 describes the proposed evolutionary algorithm, Section 4 presents the experimental study and Section 5 presents and discusses the results. Finally, Section 6 ends with conclusions.

2. Related work

The traditional single-label classification task aims to predict the class or group associated to each of the instances described by a set or input features. Each of the instances is classified in just one class from a previously defined set of classes. However, in MLC, each instance may be labeled with more than one of the q class labels simultaneously. Given a set of q predefined labels $\mathcal{L} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$, the subset of relevant labels associated with each of the instances can be viewed as a binary vector $\boldsymbol{y} = \{0,1\}^q$ where each element is 1 if the label is relevant and 0 otherwise. In this way, the goal of MLC is to predict, for an unseen instance, a bipartition including its sets of relevant ($\hat{\boldsymbol{y}}$) and irrelevant labels ($\bar{\boldsymbol{y}}$).

Several methods for MLC have been proposed in the literature, aiming to handle with the three main problems in MLC, such as the imbalance of the output space, the relationship among labels and the high dimensionality of the output space. These methods are categorized into three main groups: problem transformation, algorithm adaptation, and EMLCs [14,15].

Problem transformation methods transform the multi-label problem into one or more single-label problems. These problems are then solved by using traditional single-label classification methods. For ease of understanding, schemes of the main transformations are presented in Fig. 1. Binary Relevance (BR) [16] decomposes the multi-label problem into q independent binary single-label problems, then building q independent binary classifiers, one for each label. BR is simple and intuitive,

Original)

x	λ_1	λ_2	λ_3
x_1	0	1	0
x_2	1	0	1
x_3	0	1	1
x_4	0	1	0
x_5	1	0	1
x_6	1	1	0

LP)

x	Class
x_1	C_{010}
x_2	C_{101}
x_3	C_{011}
x_4	C_{010}
x_5	C_{101}
x_6	C_{110}

PS)

x	Class
x_1	C_{010}
x_2	C_{101}
x_3	C_{010}
x_4	C_{010}
x_5	C_{101}
x_6	C_{010}

BR)

x	λ_1
x_1	0
x_2	1
x_3	0
x_4	0
x_5	1
x_6	1

x	λ_2
x_1	1
x_2	0
x_3	1
x_4	1
x_5	0
x_6	1

x	λ_3
x_1	0
x_2	1
x_3	1
x_4	0
x_5	1
x_6	0

CC)

x	λ_1
x_1	0
x_2	1
x_3	0
x_4	0
x_5	1
x_6	1

→

x	λ_2
$x_1 \cup \hat{\lambda_1}$	1
$x_2 \cup \hat{\lambda_2}$	0
$x_3 \cup \hat{\lambda_3}$	1
$x_4 \cup \hat{\lambda_4}$	1
$x_5 \cup \hat{\lambda_5}$	0
$x_6 \cup \hat{\lambda_6}$	1

→

x	λ_3
$x_1 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	0
$x_2 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	1
$x_3 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	1
$x_4 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	0
$x_5 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	1
$x_6 \cup \hat{\lambda_1} \cup \hat{\lambda_2}$	0

Fig. 1. Main problem transformations in MLC. For PS, labelsets appearing less than 2 times are pruned and reintroduced with most frequent subsets.

but the fact of considering the labels independently makes it unable to model the compound dependencies among the labels. BR do not deal with any of the previously described problems in MLC. In order to overcome the label independence assumption of BR, Classifier Chain (CC) [5] generates q binary classifiers but linked in such a way that each binary classifier also includes the label predictions of previous classifiers in the chain as additional input features. In this way and unlike BR, CC is able to model the relationships among the labels without introducing more complexity. However, although it deal with the relationship among labels, it does not consider them, or any other characteristics of the data to select the chain. Since the order of the chain has a determinant effect on its performance, other approaches have been proposed to select the best chain ordering [17,18].

Label Powerset (LP) [19] transforms the multi-label problem into a multi-class problem, creating a new class for each distinct combination of labels, called labelset, that appears in the dataset. This method is able to strongly model the relationships among the labels, but its complexity grows exponentially with the number of labels; it is also not able to predict a labelset that does not appear in the training set. Therefore, although it is able to handle with the relationship among labels, LP greatly increases the dimensionality of the output space, as well as its imbalance. Pruned Sets (PS) [7] tries to reduce the complexity of LP, focusing on most important combinations of labels by pruning instances with less frequent labelsets. To compensate for this loss of information, it reintroduces the pruned instances with a more frequent subset of labels. Thus, PS considers the imbalance of LP's output space to reduce its dimensionality and complexity. ChiDep [20] creates groups of dependent labels based on the χ^2 test for labels dependencies identification. For each group of dependent labels it builds a LP classifier, while for each single label which is not in any group it builds a binary classifier. ChiDep tries to reduce the disadvantages of the independence assumption of the binary methods and allows for simpler LP methods. Besides, ChiDep considers the relationship among group of labels and the dimensionality of the output space in building phase, therefore being able to reduce the imbalance in each model if the groups are small.

The methods in the algorithm adaptation group adapt or extend existing machine learning methods to directly handle multi-label data. Predictive Clustering Trees (PCTs) [21] are decision trees where the data is partitioned in each node using a clustering algorithm. In order to adapt them to MLC, the distance between two instances for the clustering algorithm is calculated as the sum of the Gini Indices [22] of all labels, so it considers the relationship among labels when building the model. Instance-based algorithms have been also adapted for MLC, such as Multi-Label k-Nearest Neighbors (ML-kNN) [23]. For each unknown instance, first the k nearest neighbors are found, then the number of neighbors belonging to each label are counted, and finally the *maximum a posteriori* principle is used to identify the labels for the given instance. As ML-kNN considers all label assignments of k-nearest neighbors to label a new instance, it implicitly consider the relationship among labels to build the model. On the other hand, the traditional feed-forward neural network have been also adapted in the Back-Propagation for Multi-Label Learning (BP-MLL) [24]. In this way, an error function for multi-label scenarios was proposed, which takes into account the predicted ranking of labels. The ranking of labels also imply the relationship among labels, so BP-MLL considers it in the building phase. A wider description of algorithm adaptation methods in MLC can be found in [14].

The third group of methods includes the EMLCs. Although many of the MLC algorithms are based on the combination of several classifiers, only are considered as EMLCs those that combine several classifiers which are able to deal with multi-label data [11]. Thus, although BR combines several classifiers it is not an EMLC since it combines single-label but not multi-label classifiers. Ensemble of BR classifiers (EBR) [5] builds an ensemble of n BR classifiers where each is trained with a sample from the training dataset, being n the number of desired multi-label classifiers in the ensemble. The selection of instances in each BR provides diversity to the ensemble, but as BR, it still does not consider any of the characteristics of the data to build the model. Ensemble of Classifier Chains (ECC) [5] builds an ensemble of n CCs, each with a random chain and a random sample with replacement from the training dataset. The selection of several different chains reduces the risk of selecting a bad chain which could lead to a bad performance, however, they are all created randomly and not based on any of the characteristics of the data. Multi-Label Stacking (MLS) [25] is composed of two phases. In the first phase, q BR classifiers are learned, one for each label; while in the second phase, the input feature set is augmented with the predictions of each binary classifier from the first phase, training q new binary classifiers using the desired outputs as targets. MLS is able to model the relationship among labels thanks to the use of the predictions in the first phase to predict the labels in the second phase. Ensemble of Pruned Sets (EPS) [7] makes an ensemble of n PSs where each classifier is trained with a sample of the training set without replacement. The use of many PSs with different data subsets avoids overfitting effects of pruning instances, but as PS, in datasets with a high number of labels the complexity can be still very high.

Hierarchy Of Multi-label classifiERs (HOMER) [6] generates a tree of multi-label classifiers, where the root contains all labels and each leaf represents one label. At each node, the labels are split with a clustering algorithm, grouping similar labels into a meta-label. HOMER considers the relationship among labels to build the model, making it able to handle with smaller subsets of labels in each node, so that the dimensionality of the output space in each of them is reduced, also reducing the imbalance depending on the internal multi-label classifier used. Random Forest of Predictive Clustering Trees (RF-PCT) [26] builds an ensemble of n PCTs by selecting a random subset of the instances in each model. Further, each PCT selects at each node of the tree the best feature from a random subset of the original ones. As PCT, it considers the relationship among labels in the building phase. Finally, RAndom k-labELsets (RAkEL) [8] builds an ensemble of LP classifiers, where each is built over a random projection of the output space. In this way, RAkEL deals with the relationship among labels as LP does but in a much simpler way. RAkEL handles with the three main problems of the MLC: it is able to detect the compound dependencies among labels, it reduces the dimensionality of the output space by selecting small subsets of labels (a.k.a. k-labelsets), and also the imbalance of each of the methods is not usually high since the reduced number of labels in each of them. However, RAkEL selects the k-labelsets randomly, without considering any the characteristics of the data, which could lead to a poor performance.

A summary of the previously defined method is available in Table 1. This table indicates if each method deals with (D) and/or considers each

Table 1
Summary of state-of-the-art MLC methods. It is indicated with a 'D' if the method is able to deal with the corresponding problem (imbalance, relationships among labels, and high dimensionality of the output space), and with a 'B' if it considers this characteristic at building phase.

	Imbalance	Relationships	Output Dim.
BR			
CC		D	
LP		D	
PS	D, B	D	D, B
ChiDep	D	D, B	D, B
PCT		D, B	
MLkNN		D, B	
BP-MLL		D, B	
EBR			
ECC		D	
MLS		D	
EPS	D, B	D	D, B
HOMER	D	D, B	D, B
RF-PCT		D, B	
RAkEL	D	D	D, B

of the characteristics of the data at building phase (B). Note that there are methods that are able to deal with any of the problems, but they do not consider the corresponding characteristics when building the model. For example RA*k*EL is able to model the relationship among labels, but it does not consider neither these relationships nor other of the characteristics of the data to select the *k*-labelsets, it simply creates them randomly. On the other hand, HOMER considers the relationship among labels when building the model, since it split the labelsets into smaller ones considering the relationship among the labels.

3. Evolutionary multi-label ensemble

In this section the evolutionary algorithm is presented, focusing on the encoding scheme, the fitness function, and the genetic operators. Then, the time complexity of EME is also presented.

3.1. Evolutionary algorithm

The evolutionary algorithm is based on a generational elitist algorithm [27,28], that is, it ensures that the best individual in the last generation is the best individual of the evolution. Each individual of the evolutionary algorithm represents a full multi-label ensemble consisting of *n* multi-label classifiers, each of them modeling a *k*-labelset.

Fig. 2 presents a flowchart of the evolutionary algorithm. At the beginning, the population *p* of *popSize* individuals is randomly created, considering the size of the *k*-labelsets and the number of classifiers in each individual (Section 3.2). Then, the initial population is evaluated using the multi-label dataset (Section 3.3). In each generation, *popSize* individuals are selected by tournament selection and stored in *s*. Each individual in *s* is considered for crossover and mutation based on their respective probabilities (Section 3.4). Once the genetic operators are applied, these new individuals are evaluated. To maintain elitism, the population in each generation keeps all new individuals, unless the best parent is better than all the children; in this case, the parent replaces the worst child. The *best(set)* and *worst(set)* methods return the best and the worst individual of a *set* respectively. At the end of the generations, the best individual in the last generation is returned as the best ensemble.

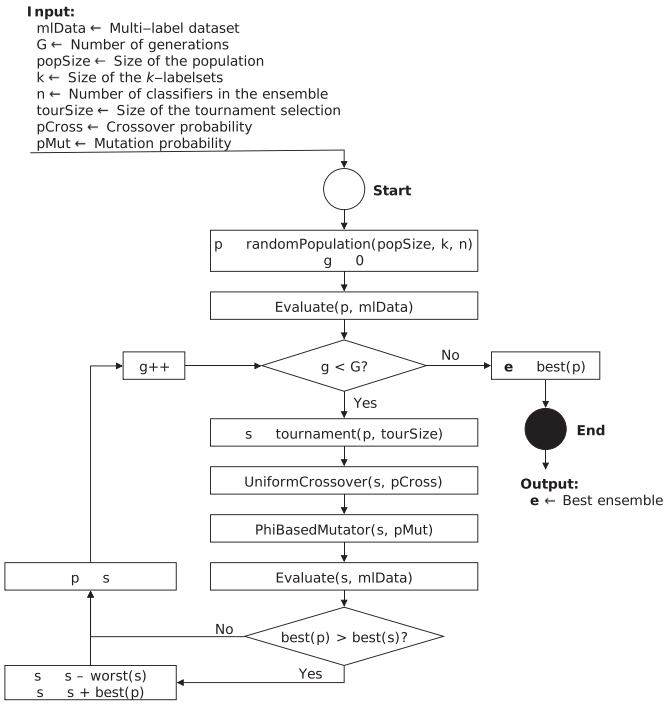


Fig. 2. Flowchart of the evolutionary process.

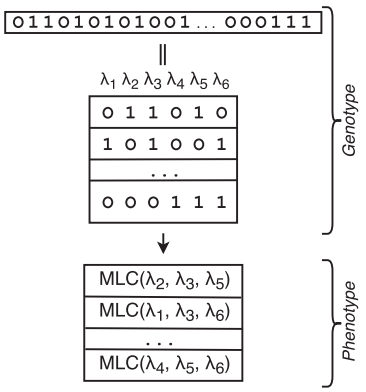


Fig. 3. Genotype and phenotype of an individual.

3.2. Individuals

The individuals, which are codified as binary arrays of $n \times q$ elements, represent a full multi-label ensemble formed by n multi-label classifiers and q labels. Each classifier of the ensemble is based on projections of the output space, built over a small subset of k labels. The parameter k is the same for all classifiers, so the number of labels in each classifier is fixed. Each fragment of size q in the individual represents the k-labelset of each classifier in the ensemble.

EME is implemented with the ability to use any multi-label classifier. However, LP is proposed as base classifier. Since EME has been designed to consider the dependencies among labels avoiding a high complexity, using a base classifier which does not consider the relationship among labels, such BR, makes no sense. Further, the use of LP has many advantages over other methods that also consider the relationship among labels, as for example CC. If k is small, as proposed, LP builds an unique model for each k-labelset able to model the dependencies among all labels at a time with a low complexity due to the reduced output space. On the other hand, CC needs to build k different binary models, and not all dependencies are considered in each model; for example the first label in the chain is modeled without considering the dependencies with the rest, the second is modeled considering the dependency with only the previous one, and so on. Besides, the use of CC would introduce a higher computational cost due to the increase in the number of different possible individuals by the different chains. The performance of EME, as well as that of the vast majority of multi-label methods, is biased by the performance of the single-label method used. Many ensemble methods in MLC have used decision trees as base classifier [5,8,29] with promising results, so the C4.5 decision tree (Weka's J48 [30]) is used as a single-label classifier. For the parameters of C4.5, we used a minimum number of objects per leaf of 2, and a pruning confidence of 0.25. Although we used these parameters, optimizing them for each specific problem would lead to a better performance, both in EME and in any other method that used C4.5.

The individuals in the initial population are generated by randomly choosing k bits to a value of 1 for each fragment representing a multi-label classifier. Then, with the evolutionary algorithm the individuals are crossed and mutated, evolving towards a more promising combination of multi-label classifiers, instead of being mere random selections. Fig. 3 shows the genotype (represented as a one-dimensional array and as a matrix) and the phenotype of an individual. For example, the first, represented by [0, 1, 1, 0, 1, 0] indicates that labels λ_2, λ_3 and λ_5 are included in the first classifier of the ensemble.

At the time of evaluating each individual, first the corresponding multi-label ensemble must be generated. For each fragment of size q in the individual, the full training dataset is filtered keeping only the labels in its k-labelset, and the corresponding multi-label classifier is built over the filtered dataset. Then, for an unknown instance, each classifier of

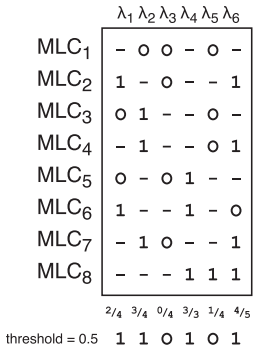


Fig. 4. Example of the voting process of the ensemble (prediction threshold = 0.5).

the ensemble provides prediction for the labels on its own k-labelset, as shown in Fig. 4. In the example in Fig. 3, the first classifier included labels λ_2, λ_3 and λ_5, so in Fig. 4 the first classifier gives a prediction for only those labels. Finally, the ratio of positive predictions for each label is calculated. If this ratio is greater than or equal to a given threshold (in the example, threshold = 0.5), the final prediction is 1 (relevant label) and 0 (irrelevant) otherwise. As seen in Fig. 4 for a certain example, label λ_5 obtains one of four possible votes, so the final prediction is 0, while for label λ_6, which obtains four of five positive votes, the final prediction is 1.

3.3. Fitness function

The fitness function measures both the performance of the classifier and the number of times that each label appears in the ensemble, thus leading the evolution towards high-performing individuals that also consider all labels the same number of times regardless of their frequency.

Many evaluation measures for MLC have been proposed in the literature, some of them identified as non-decomposable measures [31]. The non-decomposable measures evaluate the multi-label prediction as a whole, unlike others that evaluate the prediction for each label separately. As one of the objectives of EME is to consider the relationship among the labels in the ensemble, a evaluation measures which also considers this fact is used. The Example-based FMeasure (ExF) is an approach to calculate the widely used FMeasure for MLC, and it is defined in Eq. (1). The ExF is calculated for each instance, and then, the value is averaged among all the instances. ExF is defined in the range [0, 1]; the higher the value the better the performance of the algorithm. In the following, ↓ and ↑ indicate if the measures are minimized or maximized respectively.

$$\uparrow \text{ExF} = \frac{1}{m}\sum_{i=1}^{m}\frac{2|\hat{Y}_i \cap Y_i|}{|\hat{Y}_i| \cup |Y_i|} \tag{1}$$

Further, a coverage ratio measure (c_r), which evaluates the number of times that each label appears in the ensemble, has been defined. This measure is shown in Eq. (2), being v the vector of votes, i.e. a vector storing the number of times that each label appears in the ensemble, v_w a vector of votes in the worst case, and $stdv(v)$ the standard deviation of the vector v. The worst case is the one where the vector of votes is as imbalanced as possible, i.e., some labels appearing in all classifiers and the rest not being present at all. In the case where all labels appears the same number of times in the ensemble, the vector of votes is homogeneous and the standard deviation is 0. Therefore, c_r is to be minimized. The coverage ratio is divided by the worst case in order to have a measure in the range [0, 1]. If c_r were not taken into account in the fitness, labels that are easier to predict would tend to appear more frequently in the individuals, causing others barely appearing.

$$\downarrow c_r = \frac{stdv(v)}{stdv(v_w)} \tag{2}$$

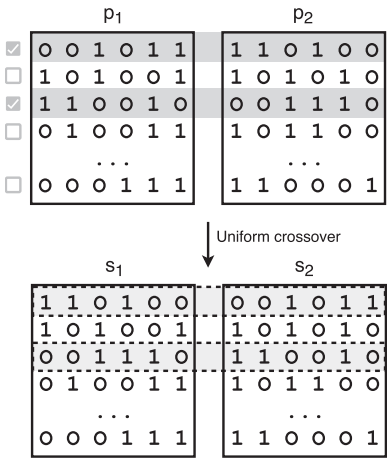


Fig. 5. Uniform crossover operator.

As an example, c_r for the case in Fig. 4 is shown in Eq. (3):

$$c_r = \frac{stdv(4,4,4,3,4,5)}{stdv(8,8,8,0,0,0)} = 0.1443 \tag{3}$$

Since both measures are in the range [0, 1], but ExF is maximized and c_r is minimized, the fitness function is defined as the linear combination of them, as shown in Eq. (4).

$$\uparrow fitness = \frac{\text{ExF} + (1 - c_r)}{2} \tag{4}$$

As all the multi-label ensembles of the population must be generated to calculate their fitness, evaluation is the process that consumes the most time. In order to reduce the runtime of the algorithm, two structures are created: one storing the fitness of each evaluated individual and other storing each multi-label classifier that was built. Thus, if an individual appears more than once, regardless of the order of its multi-label classifiers, the fitness is directly obtained from this structure, avoiding to evaluate a full ensemble. Further, if an individual which is going to be built contains a classifier that was previously built for other individual, this multi-label classifier does not have to be built again but it is directly obtained from the structure.

3.4. Genetic operators

In this section the crossover and mutation operators used in the evolutionary algorithm are described. Tournament selection is used to determine the individuals that form the set of parents. Then, each of these individuals is crossed or mutated based on crossover and mutation probabilities. The crossover and mutation operators are not mutually exclusive, i.e., an individual could be crossed and mutated in the same generation.

3.4.1. Uniform crossover operator

The uniform crossover operator swaps fragments of genotype of size q corresponding to multi-label classifiers between two parents. For each of the n fragments, the operator decides based on a probability (by default, 0.5) if the fragments in the same position in both parents are swapped. Fig. 5 shows an example of the crossover operator, where the first and third classifiers are swapped between the parents. This operator makes each ensemble explore new combinations of classifiers that were already present in other individuals. The new individuals will always be valid, because neither the number of active bits of each classifier nor the number of classifiers are modified.

3.4.2. Phi-based mutation operator

We have proposed a phi-based mutation operator as a contribution of the paper. This operator swaps two bits of different value for each fragment corresponding to a base classifier in an individual, making each

Fig. 6. Phi-based mutation operator.

classifier of the individual cease to classify one label to classify other. The bit swapping is performed considering the relationships among the labels, favoring the mutation to combinations of more related labels. In order to evaluate the relationships among the labels, the phi (ϕ) coefficient [32] that identifies the relationship between label pairs, is used. The phi coefficient is in the range $[-1, 1]$, 1 meaning total direct correlation, -1 total indirect correlation, and 0 no correlation.

Fig. 6 shows an example of the phi-based mutation operator for a fragment of an individual. First, a random position corresponding to an active label is randomly selected (Fig. 6a). Then, mutation weights w_b of each position b corresponding to each inactive label are calculated as shown in Eq. (5). The weights are calculated by accumulating the values of ϕ between the corresponding labels and each label in A, being A the set of remaining active labels (Fig. 6b). As the purpose is to evaluate the dependencies among the labels, regardless of whether positive or negative, the absolute value of phi is used. Also, a small value of ε is used to assign a small probability of mutating to the labels that are not correlated with the other active labels.

$$w_b = \varepsilon + \sum_{l \in A} |\phi_{b,l}| \tag{5}$$

Based on these weights, one of the inactive labels is selected to mutate (Fig. 6c), where labels with a higher weight are more likely to be selected. Finally, the two selected positions are swapped (Fig. 6d). Thereby, subsets of more related labels are more likely to be selected, but also keeping a small probability to search for less related combination of labels. The mutated individuals are always valid, because the number of active bits remains constant.

3.5. Time complexity

As previously stated, the most consuming process of the whole evolutionary algorithm is the evaluation of the individuals, since it requires to build each of the multi-label classifiers. The individuals are based on the use of C4.5 classifier, which complexity is $O(m \times d^2)$ [33], where m is the number examples and d is the number of features of the dataset. The complexity of EME is upper bounded by the total number of C4.5 classifiers that it has to evaluate. In each of the G generations, a total of *popSize* individuals, each composed by n C4.5 classifiers are evaluated. However, each base classifier that has been ever built is stored and EME does not have to build it again to evaluate an individual, so the number of classifier to build is usually drastically reduced. Besides, note that the number of possible C4.5 classifier to build is the same as the number of possible combinations of k labels given q. Thereby, the time complexity of EME is upper bounded by $O(m \times d^2 \times n_T)$, being n_T the number of C4.5 classifiers that could be build, defined as $n_T = \min(n \times popSize \times G, \binom{q}{k})$. Nevertheless, this asymptotic time complexity is reduced in the reality, since each individual that appears repeated in the population in any generation, is not evaluated and its fitness is directly obtained, avoiding to build an entire individual. Also note that the complexity of EME is directly related to the complexity of the base classifier used; using a different single-label classifier its complexity would vary.

Table 2
Datasets and their characteristics.

Dataset	Domain	m	q	d	*card*	*dens*	Ref.
Emotions	Audio	72	6	72	1.868	0.311	[6]
Reuters1000	Text	294	6	1000	1.126	0.188	[35]
Guardian1000	Text	302	6	1000	1.126	0.188	[35]
Bbc1000	Text	352	6	1000	1.125	0.188	[35]
3s-inter3000	Text	169	6	3000	1.142	0.190	[35]
Gnegative	Biology	1392	8	440	1.046	0.131	[36]
Plant	Biology	948	12	440	1.080	0.089	[36]
Water-quality	Chemistry	1060	16	14	5.072	0.362	[37]
Yeast	Biology	2417	14	103	4.237	0.303	[38]
Human	Biology	3108	14	440	1.190	0.084	[36]
Birds	Audio	645	19	260	1.014	0.053	[39]
Slashdot	Text	3782	22	1079	1.180	0.053	[40]
Genbase	Biology	662	27	1186	1.252	0.046	[41]
Medical	Text	978	45	1449	1.245	0.028	[42]

4. Experimental studies

The purpose of the experimental studies is to compare EME to other state-of-the-art algorithms in multi-label classification over a wide range of datasets and evaluation measures. In this section the multi-label datasets and the evaluation measures used in the experiments are first presented, and then, the experimental settings are explained.

4.1. Datasets

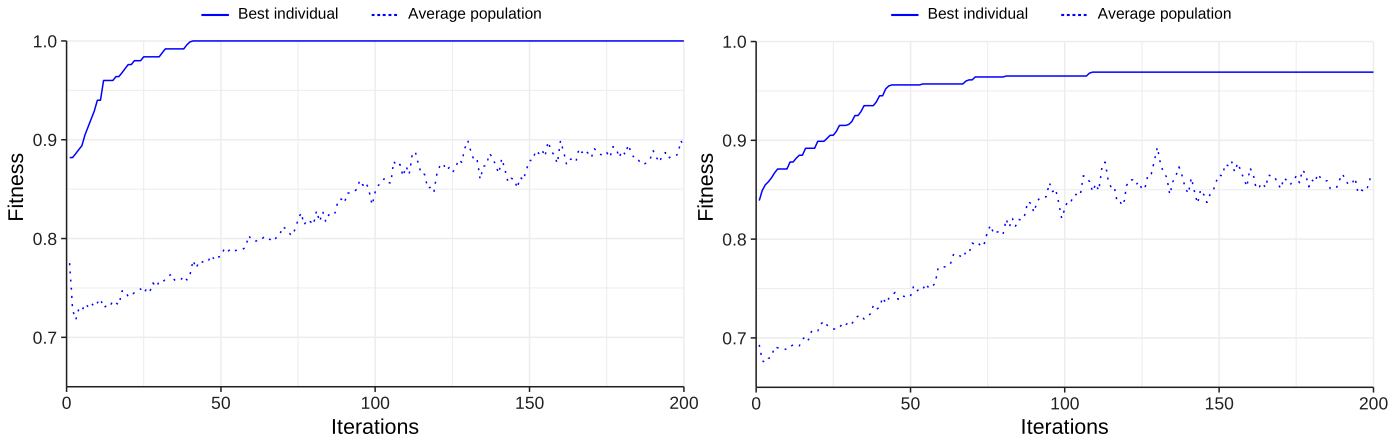
The experiments were performed over a wide set of 14 reference datasets[1] from different domains, such as text categorization, multimedia, chemistry and biology. Table 2 lists the datasets along with their main characteristics, such as domain, number of instances (m), number of labels (q), number of features (d), cardinality (*card*, mean number of labels per instance) and density (*dens*, cardinality divided by the number of labels). The datasets are ordered by number of labels. The MLDA tool [34] was used for the characterization of the datasets.

4.2. Evaluation measures

In order to evaluate multi-label classification methods, many evaluation measures that take into account all labels were proposed [43]. We have based on the study of correlation among evaluation measures carried out in [44] to select the measures for the experiments.

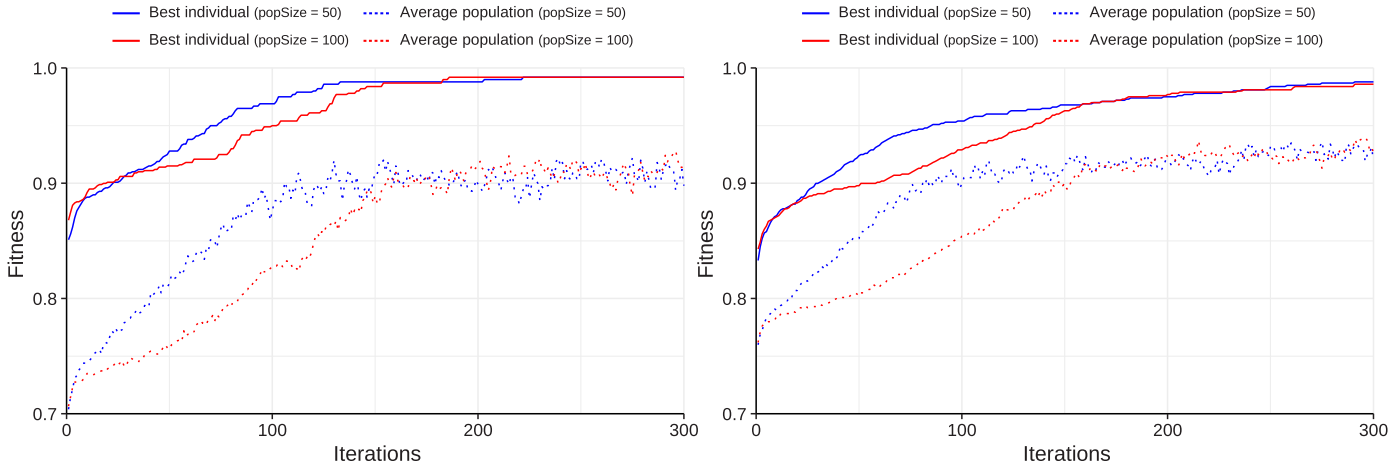
Given its wide use in the evaluation of MLC methods in the literature, Hamming loss (HL) has been selected. It is a minimized measure which computes the average number of times that a label is incorrectly predicted. HL is defined in Eq. 6, being Δ the symmetric difference between two binary sets. Subset Accuracy (SA) is a very strict evaluation measure which requires the full multi-label prediction, including relevant and irrelevant labels, to be correctly predicted. SA is defined in Eq. 7, where $[\![\pi]\!]$ returns 1 if predicate π is true and 0 otherwise. On the other hand, usually the labels that are most important, interesting or difficult to predict are the minority labels. Therefore, the macro approach, which gives the same importance to all labels in the evaluation has been selected to calculate label-based measures such as precision (MaP), recall (MaR) and specificity (MaS). These measures are formally described in Eqs. (8)–(10) respectively, being tp_i, tn_i, fp_i, and fn_i the true positives, true negatives, false positives, and false negatives for the ith label. Precision and recall are both based on measuring the prediction of relevant labels. The study in [44] did not consider the specificity measure; however, we consider that measuring the ratio of correctly

[1] All the datasets and their descriptions are available at the repository in http://www.uco.es/kdis/mllresources/.



(a) Scene (b) Flags

Fig. 7. Variation in fitness of the best individual and the average value of fitness of the population for Scene and Flags datasets.



(a) PlantGO (b) EukaryotePseAAC

Fig. 8. Variation in fitness of the best individual and the average value of fitness of the population for PlantGO and EukaryotePseAAC datasets.

predicted irrelevant labels should be also interesting.

$$\downarrow \text{HL} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{q}|Y_i \Delta \hat{Y}_i| \tag{6}$$

$$\uparrow \text{SA} = \frac{1}{m}\sum_{i=1}^{m}[\![Y_i = \hat{Y}_i]\!] \tag{7}$$

$$\uparrow \text{MaP} = \frac{1}{q}\sum_{i=1}^{q}\frac{tp_i}{tp_i + fp_i} \tag{8}$$

$$\uparrow \text{MaR} = \frac{1}{q}\sum_{i=1}^{q}\frac{tp_i}{tp_i + fn_i} \tag{9}$$

$$\uparrow \text{MaS} = \frac{1}{q}\sum_{i=1}^{q}\frac{tn_i}{tn_i + fp_i} \tag{10}$$

4.3. Experimental settings

The experimental study carried out was divided in two parts. First, EME was compared to other classic MLC methods such as BR, LP, CC, PS, and ChiDep. Then, in order to perform a more complete experimental study, EME was compared to other state-of-the-art EMLCs such as EBR, ECC, MLS, EPS, RA*k*EL, HOMER, and RF-PCT.

To compare the performance of the algorithms, the Friedman's test [45] was used for each evaluation measure. In cases where the Friedman's test indicated that there were significant differences in the performance of the algorithms with a 95% confidence, the Holm's post-hoc test [46] for comparisons of multiple classifiers involving a control method was performed. The adjusted p-values were used in the analysis, since they consider the fact of performing multiple comparisons without a significance level, providing more statistical information [47].

The experiments were carried out using a random 5-fold cross-validation and using 10 different seeds for those which use random numbers, such as CC, EBR, ECC, EPS, RA*k*EL, and RF-PCT. The default parameters, as originally recommended by their authors, were used in different algorithms employed. All methods use C4.5 as a single-label classifier. PS prunes the instances with labelsets occurring less than 3 times, and keeps the top two best ranked subsets when reintroducing the pruned instances. EPS is composed of 10 classifiers and sampling is done without replacement, keeping the rest of parameters as PS. RA*k*EL is composed of $2q$ classifiers and each with a subset of $k = 3$ labels. Both EBR and ECC are composed of 10 classifiers and use sampling with replacement. HOMER generates 3 clusters at each node and uses the *balanced k-means* clustering method. RF-PCT uses 10 trees in the ensemble, each with the full set of the training instances.

For C4.5 decision tree we used a minimum number of objects per leaf of 2, and a pruning confidence of 0.25. It should be noted that if the

Table 3
Results of classic MLC algorithms for HL ↓ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	BR	LP	CC	PS	ChiDep
Emotions	**0.220 ± 0.016**	0.254 ± 0.022	0.263 ± 0.016	0.262 ± 0.019	0.273 ± 0.016	0.252 ± 0.017
Reuters1000	**0.229 ± 0.013**	0.257 ± 0.004	0.268 ± 0.019	0.284 ± 0.022	0.276 ± 0.025	0.257 ± 0.004
Guardian1000	**0.228 ± 0.015**	0.265 ± 0.030	0.279 ± 0.021	0.287 ± 0.018	0.274 ± 0.014	0.265 ± 0.030
Bbc1000	**0.216 ± 0.016**	0.263 ± 0.015	0.264 ± 0.013	0.284 ± 0.016	0.270 ± 0.010	0.267 ± 0.017
3s-inter3000	**0.265 ± 0.014**	0.308 ± 0.026	0.312 ± 0.009	0.311 ± 0.030	0.314 ± 0.030	0.308 ± 0.026
Gnegative	**0.091 ± 0.007**	0.120 ± 0.011	0.119 ± 0.007	0.122 ± 0.009	0.118 ± 0.011	0.123 ± 0.011
Plant	**0.102 ± 0.004**	0.139 ± 0.008	0.141 ± 0.006	0.141 ± 0.005	0.144 ± 0.005	0.139 ± 0.006
Water-quality	**0.299 ± 0.007**	0.310 ± 0.007	0.375 ± 0.008	0.334 ± 0.009	0.337 ± 0.011	0.315 ± 0.012
Yeast	**0.210 ± 0.007**	0.249 ± 0.007	0.283 ± 0.006	0.268 ± 0.008	0.279 ± 0.007	0.274 ± 0.007
Human	**0.090 ± 0.002**	0.121 ± 0.002	0.126 ± 0.002	0.122 ± 0.003	0.123 ± 0.002	0.121 ± 0.001
Birds	**0.047 ± 0.004**	0.052 ± 0.010	0.063 ± 0.004	0.052 ± 0.008	0.054 ± 0.006	0.052 ± 0.010
Slashdot	**0.041 ± 0.001**	0.043 ± 0.001	0.054 ± 0.001	0.052 ± 0.007	0.053 ± 0.001	0.042 ± 0.001
Genbase	**0.001 ± 0.001**	**0.001 ± 0.001**	0.002 ± 0.001	**0.001 ± 0.000**	0.004 ± 0.001	**0.001 ± 0.001**
Medical	**0.010 ± 0.001**	0.011 ± 0.001	0.013 ± 0.002	0.010 ± 0.001	0.013 ± 0.002	0.010 ± 0.001

Table 4
Results of classic MLC algorithms for SA ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	BR	LP	CC	PS	ChiDep
Emotions	**0.248 ± 0.038**	0.170 ± 0.047	0.226 ± 0.034	0.218 ± 0.040	0.209 ± 0.056	0.191 ± 0.036
Reuters1000	0.111 ± 0.026	0.092 ± 0.047	**0.207 ± 0.055**	0.163 ± 0.051	0.204 ± 0.074	0.092 ± 0.047
Guardian1000	0.086 ± 0.035	0.069 ± 0.040	0.166 ± 0.050	0.147 ± 0.049	**0.192 ± 0.045**	0.069 ± 0.040
Bbc1000	0.120 ± 0.034	0.071 ± 0.037	**0.207 ± 0.040**	0.175 ± 0.039	0.202 ± 0.024	0.071 ± 0.037
3s-inter3000	0.040 ± 0.023	0.089 ± 0.041	0.094 ± 0.024	0.105 ± 0.050	**0.106 ± 0.057**	0.089 ± 0.041
Gnegative	0.487 ± 0.030	0.397 ± 0.035	**0.522 ± 0.031**	0.503 ± 0.035	0.520 ± 0.049	0.422 ± 0.038
Plant	0.113 ± 0.017	0.099 ± 0.009	**0.189 ± 0.033**	0.188 ± 0.021	0.172 ± 0.022	0.101 ± 0.013
Water-quality	0.014 ± 0.008	0.008 ± 0.006	0.005 ± 0.003	0.010 ± 0.007	**0.015 ± 0.011**	0.008 ± 0.007
Yeast	0.137 ± 0.013	0.070 ± 0.009	0.135 ± 0.014	**0.138 ± 0.014**	0.131 ± 0.007	0.113 ± 0.011
Human	0.159 ± 0.014	0.115 ± 0.012	0.175 ± 0.011	**0.192 ± 0.014**	0.187 ± 0.017	0.122 ± 0.011
Birds	**0.496 ± 0.048**	0.471 ± 0.069	0.429 ± 0.058	0.476 ± 0.049	0.462 ± 0.045	0.471 ± 0.069
Slashdot	0.323 ± 0.015	0.308 ± 0.019	0.410 ± 0.015	0.344 ± 0.027	**0.412 ± 0.021**	0.328 ± 0.019
Genbase	**0.966 ± 0.015**	0.965 ± 0.015	0.965 ± 0.016	0.965 ± 0.013	0.937 ± 0.019	0.965 ± 0.015
Medical	0.649 ± 0.037	0.635 ± 0.045	0.661 ± 0.044	0.664 ± 0.040	**0.666 ± 0.029**	0.665 ± 0.035

Table 5
Results of classic MLC algorithms for MaP ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	BR	LP	CC	PS	ChiDep
Emotions	**0.657 ± 0.039**	0.596 ± 0.041	0.569 ± 0.029	0.578 ± 0.027	0.560 ± 0.032	0.593 ± 0.022
Reuters1000	0.235 ± 0.068	0.215 ± 0.051	**0.287 ± 0.070**	0.211 ± 0.073	0.256 ± 0.106	0.215 ± 0.051
Guardian1000	**0.284 ± 0.088**	0.205 ± 0.053	0.221 ± 0.053	0.230 ± 0.074	0.248 ± 0.048	0.205 ± 0.053
Bbc1000	**0.362 ± 0.061**	0.262 ± 0.045	0.293 ± 0.058	0.244 ± 0.079	0.291 ± 0.043	0.256 ± 0.050
3s-inter3000	0.107 ± 0.048	0.167 ± 0.086	0.174 ± 0.038	**0.177 ± 0.064**	0.154 ± 0.055	0.167 ± 0.086
Gnegative	**0.509 ± 0.104**	0.317 ± 0.029	0.366 ± 0.125	0.316 ± 0.040	0.335 ± 0.056	0.300 ± 0.027
Plant	**0.183 ± 0.046**	0.142 ± 0.021	0.144 ± 0.014	0.142 ± 0.028	0.123 ± 0.036	0.143 ± 0.016
Water-quality	**0.558 ± 0.023**	0.521 ± 0.027	0.446 ± 0.014	0.500 ± 0.024	0.354 ± 0.048	0.510 ± 0.024
Yeast	**0.510 ± 0.029**	0.403 ± 0.009	0.377 ± 0.021	0.394 ± 0.014	0.377 ± 0.012	0.381 ± 0.010
Human	**0.220 ± 0.038**	0.163 ± 0.014	0.129 ± 0.007	0.143 ± 0.011	0.132 ± 0.019	0.158 ± 0.014
Birds	0.396 ± 0.071	**0.398 ± 0.082**	0.318 ± 0.086	0.386 ± 0.090	0.318 ± 0.048	**0.398 ± 0.082**
Slashdot	**0.529 ± 0.045**	0.518 ± 0.055	0.430 ± 0.029	0.500 ± 0.053	0.434 ± 0.043	0.514 ± 0.071
Genbase	**0.929 ± 0.050**	**0.929 ± 0.056**	0.915 ± 0.061	**0.929 ± 0.050**	0.760 ± 0.084	**0.929 ± 0.056**
Medical	**0.651 ± 0.054**	0.644 ± 0.053	0.615 ± 0.059	0.646 ± 0.049	0.621 ± 0.068	0.643 ± 0.056

parameters of C4.5 were tuned for each specific case, the performance of EME should be improved. However, this improvement should be the same for the rest of state-of-the-art methods, so tuning the parameters of C4.5 is not the objective of this paper. In this way, we carry out a fair comparison among methods that use the same parameters of C4.5.

EME was implemented using JCLEC [48] and Mulan [49] frameworks, and the code is publicly available in a GitHub repository[2]. A brief study was carried out first in order to select the parameters of the evolutionary algorithm, such as the population size, number of generations, and crossover and mutation probabilities. For the parameters of the multi-label classifier in EME, they are similar to those proposed for RAkEL, each member of the ensemble has a subset of $k = 3$ labels, the ensemble is composed of $2q$ classifiers, and the prediction threshold is 0.5.

5. Results and discussion

In this section we present the experimental results. First the experimental study to select the parameters of EME is introduced, and then, the

[2] https://github.com/kdis-lab/EME.

Table 6
Results of classic MLC algorithms for MaR ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	BR	LP	CC	PS	ChiDep
Emotions	**0.592 ± 0.025**	0.547 ± 0.028	0.561 ± 0.022	0.568 ± 0.037	0.553 ± 0.023	0.571 ± 0.032
Reuters1000	0.131 ± 0.034	0.178 ± 0.062	**0.275 ± 0.072**	0.201 ± 0.053	0.235 ± 0.069	0.178 ± 0.062
Guardian1000	0.133 ± 0.038	0.132 ± 0.032	**0.223 ± 0.054**	0.196 ± 0.055	0.217 ± 0.009	0.132 ± 0.032
Bbc1000	0.164 ± 0.039	0.136 ± 0.055	**0.291 ± 0.059**	0.202 ± 0.036	0.263 ± 0.033	0.129 ± 0.057
3s-inter3000	0.078 ± 0.042	0.173 ± 0.108	**0.206 ± 0.085**	0.191 ± 0.081	0.187 ± 0.092	0.173 ± 0.108
Gnegative	0.352 ± 0.083	0.343 ± 0.078	**0.368 ± 0.104**	0.340 ± 0.067	0.327 ± 0.052	0.330 ± 0.078
Plant	0.082 ± 0.016	**0.151 ± 0.018**	0.137 ± 0.018	**0.151 ± 0.026**	0.125 ± 0.049	**0.151 ± 0.017**
Water-quality	**0.469 ± 0.018**	0.429 ± 0.023	0.451 ± 0.021	0.445 ± 0.026	0.179 ± 0.009	0.440 ± 0.027
Yeast	0.361 ± 0.008	0.384 ± 0.006	0.375 ± 0.014	0.387 ± 0.017	0.362 ± 0.017	**0.388 ± 0.011**
Human	0.095 ± 0.011	**0.162 ± 0.017**	0.130 ± 0.011	0.148 ± 0.017	0.134 ± 0.018	0.155 ± 0.017
Birds	0.228 ± 0.048	0.269 ± 0.053	**0.292 ± 0.068**	0.264 ± 0.059	0.226 ± 0.025	0.269 ± 0.053
Slashdot	0.319 ± 0.027	0.313 ± 0.031	**0.400 ± 0.024**	0.336 ± 0.029	0.398 ± 0.024	0.314 ± 0.033
Genbase	**0.934 ± 0.045**	**0.934 ± 0.050**	0.902 ± 0.059	**0.934 ± 0.045**	0.751 ± 0.080	**0.934 ± 0.050**
Medical	**0.650 ± 0.056**	0.644 ± 0.057	0.605 ± 0.062	0.645 ± 0.051	0.600 ± 0.047	0.645 ± 0.058

Table 7
Results of classic MLC algorithms for MaS ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	BR	LP	CC	PS	ChiDep
Emotions	**0.858 ± 0.014**	0.829 ± 0.022	0.811 ± 0.013	0.808 ± 0.014	0.800 ± 0.020	0.820 ± 0.011
Reuters1000	**0.914 ± 0.015**	0.865 ± 0.018	0.833 ± 0.016	0.822 ± 0.020	0.834 ± 0.019	0.865 ± 0.018
Guardian1000	**0.918 ± 0.017**	0.864 ± 0.037	0.826 ± 0.011	0.823 ± 0.016	0.833 ± 0.007	0.864 ± 0.037
Bbc1000	**0.922 ± 0.018**	0.864 ± 0.022	0.831 ± 0.009	0.824 ± 0.012	0.833 ± 0.011	0.862 ± 0.024
3s-inter3000	**0.883 ± 0.023**	0.801 ± 0.026	0.799 ± 0.009	0.795 ± 0.032	0.801 ± 0.018	0.801 ± 0.026
Gnegative	**0.961 ± 0.006**	0.922 ± 0.013	0.922 ± 0.005	0.919 ± 0.005	0.923 ± 0.007	0.917 ± 0.009
Plant	**0.968 ± 0.004**	0.916 ± 0.009	0.916 ± 0.003	0.914 ± 0.004	0.915 ± 0.001	0.916 ± 0.008
Water-quality	0.786 ± 0.016	0.782 ± 0.016	0.687 ± 0.008	0.750 ± 0.023	**0.899 ± 0.009**	0.770 ± 0.020
Yeast	**0.803 ± 0.006**	0.745 ± 0.006	0.735 ± 0.010	0.743 ± 0.013	0.743 ± 0.009	0.730 ± 0.010
Human	**0.968 ± 0.002**	0.924 ± 0.002	0.923 ± 0.001	0.924 ± 0.002	0.925 ± 0.002	0.925 ± 0.003
Birds	**0.989 ± 0.003**	0.982 ± 0.005	0.968 ± 0.004	0.982 ± 0.003	0.982 ± 0.003	0.982 ± 0.005
Slashdot	**0.992 ± 0.001**	0.991 ± 0.001	0.972 ± 0.001	0.978 ± 0.009	0.973 ± 0.001	0.991 ± 0.002
Genbase	**1.000 ± 0.000**	**1.000 ± 0.000**	0.999 ± 0.001	**1.000 ± 0.000**	0.999 ± 0.001	**1.000 ± 0.000**
Medical	**0.995 ± 0.001**	**0.995 ± 0.001**	0.993 ± 0.001	**0.995 ± 0.001**	0.994 ± 0.001	**0.995 ± 0.001**

Table 8
Friedman's test results for the comparison with classic MLC algorithms. Values in bold indicate that there exist significant differences in the performance of the algorithms at 95% confidence.

	Statistic	*p*-value
HL	45.42	**0.0000**
SA	27.61	**0.0000**
MaP	19.97	**0.0013**
MaR	8.81	0.1171
MaS	37.72	**0.0000**

Table 9
Adjusted *p*-values of the Holm's test for the comparison with classic MLC algorithms. Algorithms marked with "–" are the control algorithm in each measure and values in bold indicates that there are significant differences with the control algorithm at 95% confidence.

	EME	BR	LP	CC	PS	ChiDep
HL	–	**0.0299**	**0.0000**	**0.0001**	**0.0000**	**0.0267**
SA	≥ 0.2	**0.0003**	≥ 0.2	≥ 0.2	–	**0.0160**
MaP	–	0.0617	**0.0041**	**0.0120**	**0.0002**	**0.0128**
MaS	–	**0.0339**	**0.0000**	**0.0000**	**0.0008**	**0.0080**

analysis and discussion of the two experiments carried out are presented, including the statistical tests performed for each of them. Te supplementary material available at the KDIS Research Group webpage[3] includes tables with the detailed results of all the experiments, including those of the selection of parameters of EME, more evaluation measures and runtime for the following experiments, and the average rankings and *p*-values of statistical tests.

5.1. Selection of parameters of EME

A brief study to select the parameters of the evolutionary algorithm in EME was carried out first. For these experiments, four datasets of different size were selected: Scene, Flags, PlantGO, and EukaryotePseAAC[4].

For the study of the size of the population and the number of generations, we analyzed the fitness value of the best individual in each generation, as well as the average value of fitness of the whole population. For the smaller datasets (i.e., Scene and Flags, with 6 and 7 labels respectively), 50 individuals and a total of 200 generations were used. Fig. 7a and b shows the fitness value of the best individual and the average value of the whole population for Scene and Flags datasets respectively. For Scene dataset, we could see as the algorithm converged soon, obtaining the best value of fitness in early generations (less than 50); however, for Flags dataset the algorithm converged over the iteration 110, moment in which also the average fitness value of the population stabilized.

Secondly, for PlantGO and EukaryotePseAAC datasets, which have 14 and 22 labels respectively, the experiments were executed with 50 and 100 individuals in the population, and with a total of 300 generations, allowing enough time for the algorithm to stabilize. Fig. 8a and b shows the fitness value of the best individual and average population

[3] http://www.uco.es/kdis/eme/.

[4] All these datasets are different from those used in the rest of the experimental study, and are available at the repository in http://www.uco.es/kdis/mllresources/.

Table 10
Results of EMLCs for HL ↓ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT
Emotions	0.220 ± 0.016	**0.200 ± 0.017**	0.202 ± 0.015	0.225 ± 0.015	0.209 ± 0.012	0.253 ± 0.015	0.259 ± 0.013	0.213 ± 0.017
Reuters1000	0.229 ± 0.013	0.227 ± 0.017	0.214 ± 0.011	0.237 ± 0.014	0.231 ± 0.017	0.317 ± 0.030	0.256 ± 0.012	**0.205 ± 0.012**
Guardian1000	0.228 ± 0.015	0.225 ± 0.016	0.210 ± 0.014	0.239 ± 0.022	0.226 ± 0.016	0.288 ± 0.018	0.264 ± 0.019	**0.202 ± 0.011**
Bbc1000	0.216 ± 0.016	0.216 ± 0.012	0.202 ± 0.011	0.222 ± 0.017	0.221 ± 0.015	0.286 ± 0.016	0.257 ± 0.007	**0.197 ± 0.010**
3s-inter3000	0.265 ± 0.014	0.246 ± 0.020	0.220 ± 0.013	0.279 ± 0.019	0.243 ± 0.016	0.302 ± 0.022	0.315 ± 0.031	**0.207 ± 0.014**
Gnegative	0.091 ± 0.007	**0.082 ± 0.007**	**0.082 ± 0.007**	0.094 ± 0.007	0.086 ± 0.006	0.117 ± 0.010	0.117 ± 0.008	0.092 ± 0.006
Plant	0.102 ± 0.004	0.097 ± 0.004	**0.093 ± 0.003**	0.107 ± 0.007	0.095 ± 0.003	0.140 ± 0.005	0.137 ± 0.005	0.096 ± 0.003
Water-quality	0.299 ± 0.007	0.295 ± 0.007	**0.290 ± 0.007**	0.311 ± 0.008	0.324 ± 0.008	0.341 ± 0.015	0.337 ± 0.016	0.314 ± 0.008
Yeast	0.210 ± 0.007	0.210 ± 0.006	**0.207 ± 0.008**	0.225 ± 0.008	0.210 ± 0.007	0.263 ± 0.008	0.273 ± 0.005	0.219 ± 0.007
Human	0.090 ± 0.002	0.088 ± 0.002	**0.085 ± 0.002**	0.097 ± 0.005	0.087 ± 0.002	0.121 ± 0.004	0.118 ± 0.003	0.090 ± 0.003
Birds	0.047 ± 0.004	**0.043 ± 0.006**	**0.043 ± 0.006**	0.048 ± 0.004	0.046 ± 0.007	0.062 ± 0.009	0.049 ± 0.007	0.046 ± 0.005
Slashdot	**0.041 ± 0.001**	0.043 ± 0.002	0.042 ± 0.001	0.042 ± 0.001	0.044 ± 0.001	0.048 ± 0.002	0.043 ± 0.001	0.043 ± 0.001
Genbase	**0.001 ± 0.001**	**0.001 ± 0.000**	**0.001 ± 0.000**	**0.001 ± 0.000**	0.004 ± 0.001	**0.001 ± 0.001**	**0.001 ± 0.001**	0.046 ± 0.003
Medical	**0.010 ± 0.001**	**0.010 ± 0.001**	0.011 ± 0.001	0.011 ± 0.001	0.012 ± 0.001	0.011 ± 0.002	0.011 ± 0.001	0.025 ± 0.001

Table 11
Results of EMLCs for SA ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT
Emotions	0.248 ± 0.038	**0.297 ± 0.036**	0.274 ± 0.037	0.250 ± 0.032	0.292 ± 0.031	0.182 ± 0.041	0.186 ± 0.039	0.284 ± 0.037
Reuters1000	0.111 ± 0.026	0.064 ± 0.031	0.040 ± 0.026	**0.129 ± 0.029**	0.115 ± 0.035	0.078 ± 0.031	0.112 ± 0.010	0.045 ± 0.022
Guardian1000	0.086 ± 0.035	0.063 ± 0.030	0.037 ± 0.020	0.092 ± 0.036	**0.130 ± 0.040**	0.069 ± 0.036	0.076 ± 0.055	0.037 ± 0.026
Bbc1000	0.120 ± 0.034	0.086 ± 0.034	0.057 ± 0.024	0.134 ± 0.028	**0.142 ± 0.042**	0.102 ± 0.051	0.088 ± 0.024	0.045 ± 0.022
3s-inter3000	0.040 ± 0.023	0.050 ± 0.029	0.025 ± 0.026	0.037 ± 0.022	0.044 ± 0.035	0.042 ± 0.027	**0.077 ± 0.044**	0.033 ± 0.032
Gnegative	0.487 ± 0.030	**0.548 ± 0.032**	0.497 ± 0.031	0.493 ± 0.030	0.513 ± 0.027	0.421 ± 0.023	0.397 ± 0.037	0.470 ± 0.030
Plant	0.113 ± 0.017	**0.140 ± 0.024**	0.089 ± 0.020	0.127 ± 0.029	0.095 ± 0.018	0.094 ± 0.015	0.109 ± 0.028	0.101 ± 0.018
Water-quality	0.014 ± 0.008	**0.017 ± 0.010**	0.016 ± 0.009	0.013 ± 0.008	0.015 ± 0.009	0.004 ± 0.004	0.008 ± 0.004	0.012 ± 0.008
Yeast	0.137 ± 0.013	**0.171 ± 0.016**	0.131 ± 0.014	0.112 ± 0.015	0.168 ± 0.015	0.076 ± 0.011	0.051 ± 0.008	0.145 ± 0.014
Human	0.159 ± 0.014	**0.174 ± 0.011**	0.141 ± 0.013	0.167 ± 0.018	0.140 ± 0.013	0.105 ± 0.004	0.122 ± 0.007	0.127 ± 0.012
Birds	0.496 ± 0.048	**0.522 ± 0.054**	0.516 ± 0.055	0.490 ± 0.045	0.515 ± 0.055	0.457 ± 0.049	0.491 ± 0.053	0.503 ± 0.057
Slashdot	0.323 ± 0.015	0.330 ± 0.021	0.303 ± 0.016	0.314 ± 0.024	**0.399 ± 0.008**	0.309 ± 0.021	0.310 ± 0.020	0.252 ± 0.013
Genbase	0.966 ± 0.015	0.968 ± 0.013	0.967 ± 0.013	0.965 ± 0.014	0.937 ± 0.018	**0.970 ± 0.009**	0.967 ± 0.016	0.000 ± 0.000
Medical	0.649 ± 0.037	0.671 ± 0.030	0.650 ± 0.025	0.641 ± 0.040	**0.674 ± 0.024**	0.654 ± 0.052	0.637 ± 0.044	0.085 ± 0.038

Table 12
Results of EMLCs for MaP ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT
Emotions	0.657 ± 0.039	0.685 ± 0.033	**0.704 ± 0.034**	0.640 ± 0.032	0.673 ± 0.029	0.588 ± 0.017	0.588 ± 0.026	0.647 ± 0.033
Reuters1000	0.235 ± 0.068	0.170 ± 0.087	0.134 ± 0.090	0.243 ± 0.048	**0.244 ± 0.089**	0.165 ± 0.030	0.208 ± 0.061	0.137 ± 0.099
Guardian1000	**0.284 ± 0.088**	0.166 ± 0.070	0.133 ± 0.083	0.250 ± 0.076	0.272 ± 0.114	0.242 ± 0.049	0.202 ± 0.070	0.120 ± 0.111
Bbc1000	**0.362 ± 0.061**	0.216 ± 0.102	0.214 ± 0.123	0.353 ± 0.077	0.359 ± 0.098	0.248 ± 0.052	0.267 ± 0.069	0.179 ± 0.117
3s-inter3000	0.107 ± 0.048	0.139 ± 0.092	0.094 ± 0.090	0.144 ± 0.061	0.090 ± 0.063	**0.165 ± 0.074**	0.157 ± 0.080	0.117 ± 0.107
Gnegative	**0.509 ± 0.104**	0.495 ± 0.094	0.499 ± 0.082	0.476 ± 0.091	0.473 ± 0.094	0.369 ± 0.087	0.349 ± 0.050	0.406 ± 0.087
Plant	0.183 ± 0.046	0.178 ± 0.051	0.189 ± 0.057	**0.190 ± 0.054**	0.170 ± 0.061	0.145 ± 0.028	0.157 ± 0.023	0.135 ± 0.041
Water-quality	0.558 ± 0.023	0.556 ± 0.024	**0.573 ± 0.024**	0.536 ± 0.024	0.281 ± 0.049	0.500 ± 0.023	0.498 ± 0.029	0.522 ± 0.022
Yeast	0.510 ± 0.029	0.495 ± 0.037	**0.515 ± 0.034**	0.463 ± 0.029	0.505 ± 0.054	0.389 ± 0.014	0.392 ± 0.007	0.465 ± 0.035
Human	0.220 ± 0.038	0.222 ± 0.035	**0.224 ± 0.040**	0.206 ± 0.033	0.211 ± 0.053	0.143 ± 0.012	0.171 ± 0.018	0.182 ± 0.043
Birds	0.396 ± 0.071	0.431 ± 0.078	0.420 ± 0.082	0.398 ± 0.086	0.330 ± 0.066	0.318 ± 0.049	0.408 ± 0.127	**0.432 ± 0.078**
Slashdot	0.529 ± 0.045	0.522 ± 0.044	0.521 ± 0.051	0.524 ± 0.045	**0.535 ± 0.028**	0.463 ± 0.052	0.506 ± 0.034	0.477 ± 0.024
Genbase	**0.929 ± 0.050**	0.923 ± 0.053	0.921 ± 0.053	0.925 ± 0.053	0.768 ± 0.078	0.921 ± 0.071	**0.929 ± 0.056**	0.217 ± 0.104
Medical	**0.651 ± 0.054**	0.645 ± 0.055	0.647 ± 0.063	0.645 ± 0.049	0.625 ± 0.061	0.627 ± 0.055	0.646 ± 0.057	0.379 ± 0.067

with both configurations for PlantGO and EukaryotePseAAC datasets respectively. For PlantGO, we can see that in early generations the configuration with 50 individuals achieved better fitness values, however, at the end of the evolution both configurations reached the same fitness for the best individual. On the other hand, the behavior for EukaryotePseAAC dataset is similar to PlantGO, but in this case the algorithm reached the best value in the last generations, where the configuration with 50 individuals obtained slightly better results. For both cases, the configuration with 50 individuals obtained the same or slightly better results, obtaining also a lower execution runtime.

Given these results, for datasets with a small number of labels ($\leq$ 8 labels), 50 individuals and a total of 110 generations were used. Further, for datasets where the label space is more complex ($>$ 8 labels) and therefore the search space is much wider, also 50 individuals and a total of 300 generations were used.

On the other hand, the probability of crossover and mutate an individual could have a direct effect on the final performance of the algorithm, since their variation would vary the diversity of the population and could lead to a premature convergence of the algorithm or to never converge; therefore, an experimental study to select the optimal values for both probabilities was performed. For that, values of $pCross = \{0.7, 0.8, 0.9\}$ and $pMut = \{0.1, 0.2, 0.3\}$ were used. For both Scene and Flags datasets, the best results were obtained with $pCross = 0.9$, and $pMut = 0.2$, so we selected this configuration for small datasets. On the other hand, EukaryotePseAAC obtained the best performance with $pCross = 0.8$ and $pMut = 0.2$, while PlantGO also obtained competitive

Table 13
Results of EMLCs for MaR ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	ECC	EBR	RA*k*EL	EPS	HOMER	MLS	RF-PCT
Emotions	0.592 ± 0.025	0.585 ± 0.033	0.590 ± 0.029	0.627 ± 0.034	0.621 ± 0.025	0.606 ± 0.048	0.575 ± 0.024	**0.667 ± 0.031**
Reuters1000	0.131 ± 0.034	0.084 ± 0.042	0.045 ± 0.026	0.158 ± 0.032	0.130 ± 0.034	0.158 ± 0.029	**0.162 ± 0.044**	0.044 ± 0.022
Guardian1000	0.133 ± 0.038	0.077 ± 0.029	0.041 ± 0.019	0.163 ± 0.040	0.142 ± 0.040	**0.226 ± 0.048**	0.141 ± 0.040	0.040 ± 0.025
Bbc1000	0.164 ± 0.039	0.086 ± 0.028	0.057 ± 0.023	0.177 ± 0.038	0.163 ± 0.039	**0.235 ± 0.052**	0.164 ± 0.032	0.039 ± 0.019
3s-inter3000	0.078 ± 0.042	0.069 ± 0.038	0.031 ± 0.026	0.098 ± 0.050	0.053 ± 0.036	0.165 ± 0.060	**0.166 ± 0.094**	0.035 ± 0.031
Gnegative	0.352 ± 0.083	0.341 ± 0.071	0.299 ± 0.067	**0.386 ± 0.087**	0.298 ± 0.060	0.374 ± 0.060	0.360 ± 0.083	0.255 ± 0.061
Plant	0.082 ± 0.016	0.069 ± 0.014	0.051 ± 0.015	0.101 ± 0.020	0.046 ± 0.013	0.140 ± 0.023	**0.165 ± 0.032**	0.042 ± 0.009
Water-quality	0.469 ± 0.018	0.519 ± 0.014	0.465 ± 0.015	0.525 ± 0.022	0.148 ± 0.011	0.579 ± 0.042	0.453 ± 0.017	**0.587 ± 0.018**
Yeast	0.361 ± 0.008	0.389 ± 0.009	0.351 ± 0.009	0.405 ± 0.017	0.358 ± 0.008	**0.406 ± 0.017**	0.394 ± 0.020	0.402 ± 0.010
Human	0.095 ± 0.011	0.086 ± 0.008	0.066 ± 0.007	0.118 ± 0.016	0.064 ± 0.006	0.141 ± 0.013	**0.157 ± 0.026**	0.057 ± 0.005
Birds	0.228 ± 0.048	0.222 ± 0.055	0.201 ± 0.048	0.246 ± 0.050	0.198 ± 0.047	**0.295 ± 0.057**	0.271 ± 0.074	0.217 ± 0.043
Slashdot	0.319 ± 0.027	0.321 ± 0.023	0.305 ± 0.022	0.317 ± 0.029	**0.399 ± 0.018**	0.325 ± 0.028	0.314 ± 0.035	0.232 ± 0.020
Genbase	0.934 ± 0.045	0.929 ± 0.052	0.926 ± 0.049	0.931 ± 0.048	0.759 ± 0.075	0.912 ± 0.065	**0.935 ± 0.050**	0.216 ± 0.104
Medical	**0.650 ± 0.056**	0.646 ± 0.054	0.641 ± 0.060	0.645 ± 0.052	0.600 ± 0.059	0.598 ± 0.063	0.646 ± 0.055	0.335 ± 0.058

Table 14
Results of EMLCs for MaS ↑ measure and standard deviations. Values in bold indicate the best results for each dataset.

	EME	ECC	EBR	RA*k*EL	EPS	HOMER	MLS	RF-PCT
Emotions	0.858 ± 0.014	0.861 ± 0.014	**0.881 ± 0.012**	0.834 ± 0.016	0.856 ± 0.012	0.805 ± 0.013	0.810 ± 0.012	0.828 ± 0.014
Reuters1000	0.914 ± 0.015	0.924 ± 0.017	0.952 ± 0.011	0.896 ± 0.014	0.910 ± 0.017	0.793 ± 0.042	0.867 ± 0.014	**0.964 ± 0.014**
Guardian1000	0.918 ± 0.017	0.929 ± 0.016	0.958 ± 0.013	0.897 ± 0.021	0.910 ± 0.017	0.822 ± 0.019	0.863 ± 0.016	**0.969 ± 0.011**
Bbc1000	0.922 ± 0.018	0.936 ± 0.015	0.964 ± 0.010	0.911 ± 0.018	0.912 ± 0.014	0.817 ± 0.020	0.865 ± 0.010	**0.974 ± 0.011**
3s-inter3000	0.883 ± 0.023	0.907 ± 0.023	0.952 ± 0.016	0.863 ± 0.024	0.918 ± 0.015	0.811 ± 0.026	0.794 ± 0.023	**0.967 ± 0.014**
Gnegative	0.961 ± 0.006	0.964 ± 0.004	**0.973 ± 0.004**	0.949 ± 0.008	0.968 ± 0.004	0.922 ± 0.007	0.922 ± 0.009	0.964 ± 0.004
Plant	0.968 ± 0.004	0.974 ± 0.004	**0.985 ± 0.002**	0.959 ± 0.012	0.982 ± 0.003	0.915 ± 0.007	0.917 ± 0.003	0.979 ± 0.004
Water-quality	0.786 ± 0.016	0.759 ± 0.018	0.800 ± 0.017	0.735 ± 0.021	**0.940 ± 0.007**	0.662 ± 0.046	0.741 ± 0.029	0.685 ± 0.018
Yeast	0.803 ± 0.006	0.774 ± 0.006	**0.804 ± 0.005**	0.761 ± 0.013	0.786 ± 0.005	0.727 ± 0.017	0.743 ± 0.010	0.746 ± 0.005
Human	0.968 ± 0.002	0.969 ± 0.002	**0.981 ± 0.002**	0.954 ± 0.008	0.975 ± 0.002	0.927 ± 0.003	0.927 ± 0.003	0.972 ± 0.003
Birds	0.989 ± 0.003	0.993 ± 0.002	**0.995 ± 0.001**	0.986 ± 0.003	0.992 ± 0.003	0.970 ± 0.006	0.985 ± 0.005	0.991 ± 0.002
Slashdot	0.992 ± 0.001	0.989 ± 0.002	0.992 ± 0.001	0.992 ± 0.001	0.985 ± 0.001	0.984 ± 0.002	0.990 ± 0.001	**0.997 ± 0.001**
Genbase	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	0.999 ± 0.001	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**
Medical	0.995 ± 0.001	0.994 ± 0.001	0.995 ± 0.001	0.995 ± 0.001	0.994 ± 0.001	0.995 ± 0.001	0.995 ± 0.001	**0.999 ± 0.001**

results with this configuration, so we selected it as default configuration for bigger datasets. The results of these experiments are fully available at the KDIS Research Group webpage[5].

5.2. Experiment 1: Comparing EME with other classic MLC algorithms

In this first experiment, EME is compared to other classic state-of-the-art MLC algorithms. The results of EME and the rest of state-of-the-art algorithms over all datasets are shown in Tables 3–7 for HL, SA, MaP, MaR, and MaS evaluation measures, respectively.

For HL, EME performed the best in all cases, including a tie with BR, CC and ChiDep for Genbase dataset. For SA, the best results are more spread, where EME, LP, CC, and PS achieved the best result in many datasets. In the case of MaP, EME again shown the best performance in 11 out of 14 datasets. MaP and MaR are opposite measures, so good results in one of them usually lead to bad results in the other; for MaR LP achieved the best performance in seven datasets, while EME was the best in four, BR and ChiDep in three each, and finally CC was the best in two datasets. MaP and MaR measures are both focused on relevant labels; on the other hand MaS measures the ratio of correctly predicted irrelevant labels. For MaS, EME performed the best in 13 out of 14 datasets, being the best method so far. Despite the opposition of evaluation measures, EME was able to achieve great performance in all of them.

The results of the Friedman's test for all evaluation measures, including the Friedman's statistics and the *p*-values are shown in Table 8. In four measures the Friedman's test determined that significant differences exists in the performance of the algorithms at 95% confidence, so the Holm's post-hoc test was also performed, and the adjusted *p*-values are shown in Table 9.

[5] http://www.uco.es/kdis/eme/.

Table 15
Friedman's test results for the comparison with state-of-the-art EMLCs. Values in bold indicate that there exist significant differences in the performance of the algorithms at 95% confidence.

	Statistic	*p*-value
HL	52.99	**0.0000**
SA	31.65	**0.0000**
MaP	26.74	**0.0004**
MaR	42.49	**0.0000**
MaS	50.84	**0.0000**

For the four evaluation measures where the Friedman's test indicated that there were significant differences in the performance of the algorithms, EME had the better performance in three of them. For HL and MaS, EME performed significantly better than the rest of methods, while for MaP it performed statistically better than all except BR. Further, for SA, where EME was not the control algorithm, it performed statistically equal than the control algorithm. These results showed that our algorithm has statistically better performance than the rest of classic MLC algorithms for all evaluation measures.

5.3. Experiment 2: Comparing EME with other EMLCs

Although EME has already shown that performs significantly better than classic state-of-the-art MLC algorithms, its performance was also compared to other state-of-the-art EMLCs. The results of EME and the rest of EMLCs over all datasets are shown in Tables 10–14, again for HL, SA, MaP, MaR, and MaS measures.

Table 16
Adjusted *p*-values of the Holm's test for the comparison among state-of-the-art EMLCs. Algorithms marked with '–" are the control algorithm in each measure and values in bold indicates that there are significant differences with the control algorithm at 95% confidence.

	EME	ECC	EBR	RA*k*EL	EPS	HOMER	MLS	RF-PCT
HL	0.1016	≥ 0.2	–	**0.0014**	**0.0436**	**0.0000**	**0.0000**	0.1016
SA	0.1613	–	**0.0081**	0.1613	≥ 0.2	**0.0013**	**0.0052**	**0.0003**
MaP	–	≥ 0.2	≥ 0.2	≥ 0.2	0.1636	**0.0007**	**0.0308**	**0.0007**
MaR	≥ 0.2	**0.0436**	**0.0001**	≥ 0.2	**0.0030**	–	≥ 0.2	**0.0002**
MaS	0.0758	0.0758	–	**0.0011**	0.0745	**0.0000**	**0.0000**	≥ 0.2

For HL, EBR performed the best in seven datasets, while ECC in five, RF-PCT in four, EME in three and both RA*k*EL, HOMER and MLS in one tied. As can be shown, the performance of EME in HL is better for datasets where the number of labels is greater. That means that when the label space is wider, EME tends to predict correctly, in average, a greater number of labels than the rest of methods. This is given by the fact that for cases where a greater number of different possible combinations of *k*-labelsets are available, EME is able to obtain a good combination of subsets of labels with a great performance. It can be also seen as EME obtained a better performance than RA*k*EL in all cases, enhancing the need for optimizing the combination of *k*-labelsets instead of only making a random selection. In the case of SA, EME did not achieved the best results in any dataset for such a strict measure, in which ECC was the best in seven datasets. SA evaluates the ratio of multi-label predictions where both the relevant and irrelevant labels were exactly predicted. Although it is an interesting evaluation measure in some cases, it must be interpreted cautiously since it does not consider partially correct predictions. For example, a method with a low value of SA could be predicting the rest of instances with almost all relevant labels, while a method with a higher value of SA could be predicting completely bad the rest of instances. For MaP, EME was the best in five datasets, followed by EBR being the best in four. ECC, which achieved better results in other measures, was not the best in any case for MaP. Further, although MaP and MaR are opposite measures, ECC did not achieve great results in MaR either, being the best in only one case. On the other hand, EME was the best in only one case in MaR but achieved better results for MaP, which is an expected behavior. Both ECC and EBR were not the best in any case for this measure. Finally, for MaS the better results were spread between EBR and RF-PCT, being the best in seven datasets each, while the rest in only one.

As in the previous experiment, first Friedman's test was performed in order to know if there were significant differences on the performance of the algorithms. The results of Friedman's test are shown in Table 15, indicating that significant differences exist for all measures at 95% confidence. Therefore, the post-hoc Holm's test was performed for all the measures. The results, including the adjusted *p*-values are shown in Table 16.

Although EME was the best performing algorithm for only one evaluation measure, it was the only one that did not have significant differences with the control algorithm in any measure. ECC and EBR, which achieved great results in some evaluation measures, being the control algorithm in one and two cases respectively, also had a significantly poor performance than the control algorithm in some cases, such as for SA and MaR. These results showed that EME is more consistent in overall performance than other state-of-the-art EMLCs over all measures, and did not perform significantly worse than the rest in any case. EME achieved high predictive performance compared not only with classic MLC algorithms, but also when compared with other EMLCs.

Further, EME had a better overall performance than RA*k*EL in four of the five measures, including HL and MaS, where RA*k*EL performed significantly worse than the control algorithm. This indicates that the fact of not only selecting the *k*-labelsets randomly as RA*k*EL does, but also evolving towards a more promising combination of *k*-labelsets in the ensemble makes the model to achieve a better predictive performance.

6. Conclusions

In this paper we presented an evolutionary algorithm for the automatic generation of ensembles of multi-label classifiers based on projections of labels, taking into account the relationships among the labels but avoiding a high complexity. Each individual in the evolutionary algorithm encodes an ensemble of multi-label classifiers, which are evaluated taking into account both the predictive performance of the individual and the number of times that each label appears in the ensemble. The evolutionary algorithm helps to obtain a promising and high-performing combination of multi-label classifiers into an ensemble.

The experiments over a wide set of fourteen datasets and five evaluation measures showed that our algorithm performed statistically better than classic MLC methods and also had a more consistent performance than other other state-of-the-art EMLCs. EME obtained the best results in several cases, and although not being always the first algorithm in the ranking, EME was the only algorithm that did not perform significantly worse than the rest in any case. Further, the experimental results also show that the fact of evolving the individuals toward more promising combinations of multi-label classifiers achieves better results than just selecting them randomly, as RA*k*EL does.

As future work, we aim to extend EME to use a variable number of labels (*k*) in each of the classifiers of the ensemble and to explore other ways to combine the predictions of the classifiers to create the final ensemble prediction. Further, we we aim to perform an optimization and tuning of the parameters of the single-label classifier in order to improve the performance of the final multi-label classifier. Finally, we aim to explore some multi-objective fitness functions that may improve the performance of EME, instead of selecting only one.

Acknowledgements

This research was supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund, project TIN2017-83445-P. This research was also supported by the Spanish Ministry of Education under FPU Grant FPU15/02948.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.inffus.2018.11.013.

References

[1] L. Tang, H. Liu, Scalable learning of collective behavior based on sparse social dimensions, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 09), 2009, pp. 1107–1116.
[2] G. Nasierding, A. Kouzani, Image to text translation by multi-label classification, in: Proceedings of the Advanced Intelligent Computing Theories and Applications with Aspects of Artificial Intelligence, volume 6216, 2010, pp. 247–254.
[3] E. Loza, J. Fürnkranz, Efficient multilabel classification algorithms for large-scale problems in the legal domain, in: Semantic Processing of Legal Texts, volume 6036, 2010, pp. 192–215.
[4] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: measures and random resampling algorithms, Neurocomputing 163 (Supplement C) (2015) 3–16.
[5] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (3) (2011) 335–359.

[6] G. Tsoumakas, I. Katakis, I. Vlahavas, Effective and efficient multilabel classification in domains with large number of labels, in: Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD08), volume 21, 2008, pp. 53–59.
[7] J. Read, A pruned problem transformation method for multi-label classification, in: Proceedings of the NZ Computer Science Research Student Conference, 2008, pp. 143–150.
[8] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multi-label classification, IEEE Trans. Knowl. Data Eng. 23 (7) (2011) 1079–1089.
[9] K. Laghmari, C. Marsala, M. Ramdani, An adapted incremental graded multi-label classification model for recommendation systems, Prog. Artif. Intell. 7 (1) (2018) 15–29.
[10] R. Sousa, J. Gama, Multi-label classification from high-speed data streams with adaptive model rules and random rules, Prog. Artif. Intell. 7 (3) (2018) 177–187.
[11] J.M. Moyano, E.L. Gibaja, K.J. Cios, S. Ventura, Review of ensembles of multi-label classifiers: models, experimental study and prospects, Inf. Fus. 44 (2018) 33–45.
[12] G. Brown, J. Wyatt, R. Harris, X. Yao, Diversity creation methods: a survey and categorisation, Inf. Fus. 6 (1) (2005) 5–20.
[13] L.I. Kuncheva, C.J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, Mach. Learn. 51 (2) (2003) 181–207.
[14] E. Gibaja, S. Ventura, Multi-label learning: a review of the state of the art and ongoing research, WIREs Data Mining Knowl Discov 2014.
[15] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Deroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognit. 45 (9) (2012) 3084–3104.
[16] G. Tsoumakas, I. Katakis, I. Vlahavas, Data Mining and Knowledge Discovery Handbook, Part 6, Springer, pp. 667–685.
[17] K. Dembczynski, W. Cheng, E. Hüllermeier, Bayes optimal multilabel classification via probabilistic classifier chains, in: Proceedings of the International Conference on Machine Learning, volume 10, 2010, pp. 279–286.
[18] E. Goncalves, A. Plastino, A.A. Freitas, Simpler is better: a novel genetic algorithm to induce compact multi-label chain classifiers., in: Proceedings of the 2015 Conference on Genetic and Evolutionary Computation Conference (GECCO-2015), 2015, pp. 559–566.
[19] G. Tsoumakas, I. Katakis, Multi-label classification: an overview, Int. J. Data Warehouse. Min. 3 (3) (2007) 1–13.
[20] L. Tenenboim-Chekina, L. Rokach, B. Shapira, Identification of label dependencies for multi-label classification, in: Proceedings of the Working Notes of the Second International Workshop on Learning from Multi-Label Data, 2010, pp. 53–60.
[21] H. Blockeel, L.D. Raedt, J. Ramon, Top-down induction of clustering trees, in: Proceedings of the Fifteenth International Conference on Machine Learning, in: ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 55–63.
[22] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Wadsworth and Brooks, 1984.
[23] M.-L. Zhang, Z.-H. Zhou, A k-nearest neighbor based algorithm for multi-label classification, in: Proceedings of the IEEE International Conference on Granular Computing (GrC), 2, The IEEE Computational Intelligence Society, Beijing, China, 2005, pp. 718–721.
[24] M.-L. Zhang, Z.-H. Zhou, Multi-label neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (2006) 1338–1351.
[25] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, I. Vlahavas, Correlation-based pruning of stacked binary relevance models for multi-label learning, in: Proceedings of the 1st International Workshop on Learning from Multi-Label Data (MLD'09), 2009, pp. 101–116.
[26] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Ensembles of multi-objective decision trees, in: Proceedings of the European Conference on Machine Learning, Springer, 2007, pp. 624–631.
[27] K. Deb, An introduction to genetic algorithms, Sadhana 24 (4) (1999) 293–315.
[28] D. Thierens, Selection schemes, elitist recombination, and selection intensity, in: Proceedings of the 7th International Conference on Genetic Algorithms, 1998, pp. 152–159.
[29] L. Rokach, A. Schclar, E. Itach, Ensemble methods for multi-label classification, Expert. Syst. Appl. 41 (16) (2014) 7507–7523.
[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[31] K. Dembczyński, W. Waegeman, W. Cheng, E. Hüllermeier, On label dependence and loss minimization in multi-label classification, Mach. Learn. 88 (1) (2012) 5–45.
[32] J. Cohen, P. Cohen, S.G. West, L.S. Aiken, Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences, Psychology Press, 2002.
[33] J. Su, H. Zhang, A fast decision tree learning algorithm, in: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1, in: AAAI'06, 2006, pp. 500–505.
[34] J.M. Moyano, E.L. Gibaja, S. Ventura, MLDA: A tool for analyzing multi-label datasets, Knowl. Based Syst. 121 (2017) 1–3.
[35] D. Greene, P. Cunningham, A matrix factorization approach for integrating multiple data views, in: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part I, in: ECML PKDD '09, 2009, pp. 423–438.
[36] J. Xu, J. Liu, J. Yin, C. Sun, A multi-label feature extraction algorithm via maximizing feature variance and feature-label dependence simultaneously, Knowl. Based Syst. 98 (2016) 172–184.
[37] H. Blockeel, S. Deroski, J. Grbovi, Simultaneous prediction of multiple chemical parameters of river water quality with tilde, Lect Notes Comput Sci 1704 (1999) 32–40.
[38] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, in: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, in: NIPS'01, 2001, pp. 681–687.
[39] F. Briggs, Y. Huang, R. Raich, K. Eftaxias, Z. Lei, W. Cukierski, S.F. Hadley, A. Hadley, M. Betts, X.Z. Fern, J. Irvine, L. Neal, A. Thomas, G. Fodor, G. Tsoumakas, H.W. Ng, T.N.T. Nguyen, H. Huttunen, P. Ruusuvuori, T. Manninen, A. Diment, T. Virtanen, J. Marzat, J. Defretin, D. Callender, C. Hurlburt, K. Larrey, M. Milakov, The 9th annual MLSP competition: New methods for acoustic classification of multiple simultaneous bird species in a noisy environment, in: Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2013, Southampton, United Kingdom, September 22–25, 2013, 2013, pp. 1–8.
[40] J. Read, Scalable Multi-Label Classification (PhD Thesis), University of Waikato (2010).
[41] S. Diplaris, G. Tsoumakas, P. Mitkas, I. Vlahavas, Protein classification with multiple algorithms, in: Proceedings of the 10th Panhellenic Conference on Informatics (PCI 2005), 2005, pp. 448–456.
[42] H. Shao, G. Li, G. Liu, Y. Wang, Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine, Sci. China Inf. Sci. 56 (5) (2013) 1–13.
[43] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv. 47 (3) (2015).
[44] R.B. Pereira, A. Plastino, B. Zadrozny, L.H. Merschmann, Correlation analysis of performance measures for multi-label classification, Inf. Process. Manag. 54 (3) (2018) 359–369.
[45] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Ann. Math. Statist. 11 (1) (1940) 86–92.
[46] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. (1979) 65–70.
[47] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (Dec) (2008) 2677–2694.
[48] S. Ventura, C. Romero, A. Zafra, J.A. Delgado, C. Hervás, JCLEC: A java framework for evolutionary computation, Softw. Comput. 12 (4) (2008) 381–392.
[49] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: a java library for multi-label learning, J. Mach. Learn. Res. 12 (2011) 2411–2414.

Chapter 5

Evolutionary approach to evolve ensemble members: EAGLET

We have shown good performance of EME. Now we aim to develop a second EA, called EAGLET, to improve some of the drawbacks of EME. Despite its great predictive performance, EME's biggest drawback is its computational complexity, due to the fact of evolving the entire ensemble as an individual, and therefore evaluating such a complex structure. Furthermore, it is harder for EME to find an optimal structure of the ensemble, since evaluating the ensemble as a whole, it is not able to discriminate if some of the base classifiers are useful and some other are decrementing the predictive performance.

The goal of EAGLET is to build an EMLC by selecting simple, accurate, and diverse multi-label classifiers. It codifies a separate multi-label classifier or future hypothetical member of the ensemble in each of the individuals. By doing this, EAGLET is able to determine the quality of each of the classifiers separately, which is useful to generate the consequent EMLC. Furthermore, the fact of dealing with separate classifiers also leads to a reduction in the computational complexity of the EA, since it drastically reduces the number of classifiers to evaluate in total.

Despite the fact that EMLCs tend to perform better than simpler methods, the selection of ensemble members is not trivial, but a key point to be carefully handled. As EAGLET deals with individual base classifiers, a procedure to build the ensemble given this pool of individuals needs to be proposed. The EMLC is generated incrementally as follows: I) the best individual of the population, according

only to its predictive performance, is selected and included in the current ensemble; II) the distance between each individual and the current ensemble is calculated in terms of the labels that are considered, i.e., individuals modeling labels that are less present in the ensemble will have a greater distance to the ensemble and then a higher chance to be included; III) the individual that maximizes a linear combination between its predictive performance and the diversity or distance to the current ensemble is selected; IV) steps II and III are repeated until the desired size of the ensemble is reached.

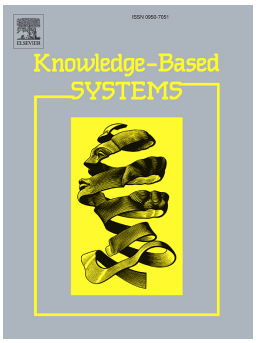
This procedure enables EAGLET to build an EMLC including not only accurate multi-label classifiers, but also classifiers that are diverse among them and considering the number of appearances of each label in the ensemble. Thus, regardless of the frequency of each labels, they are all considered in the ensemble.

As for the experimental study, a preliminary study on the parameters of the EA demonstrates that EAGLET achieves better performance when more weight is given to the selection of diverse classifiers than to the selection of more accurate members. Then, EAGLET demonstrates to significantly outperform EME, thus proving that the fact of evolving the members of the ensemble separately instead of the entire ensemble lead EAGLET to a better predictive performance. Later, EAGLET demonstrates to perform significantly better than standard MLC methods and state-of-the-art EMLCs, not only being the best method in overall for all metrics (i.e., the one with better average ranking value), but also it is the only one that do not perform significantly worse than any of the methods in any case. The performance of the rest of methods is significantly worse than the control algorithm in at least one metric each. Tables with full results are available online[1].

Finally, it is shown that the efficiency of EAGLET is much higher than EME, and also than other MLC methods based on EAs, such as GACC. Furthermore, although being more complex than other EMLCs such as ECC and RA*k*EL, given that the predictive performance of EAGLET is significantly better than those EMLCs, it is a great option in cases where the runtime of the algorithm is not a main problem.

Following, the paper associated with this chapter of the thesis [J5] is presented.

[1] https://www.uco.es/kdis/eaglet/



Title	Combining multi-label classifiers based on projections of the output space using evolutionary algorithms
Authors	Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastián Ventura
Journal	Knowledge-Based Systems
Volume	196
Article no.	105770
Year	2020
DOI	10.1016/j.knosys.2020.105770

IF (JCR 2018)	5.101
Category	Computer Science - Artificial Intelligence
Position	17/133 (Q1)
Cites (27/05/2020)	0 (WoS), 0 (Scopus), 0 (Google Scholar)

Knowledge-Based Systems 196 (2020) 105770

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Combining multi-label classifiers based on projections of the output space using Evolutionary algorithms ☆

Jose M. Moyano [a,b], Eva L. Gibaja [a,b], Krzysztof J. Cios [c,d], Sebastián Ventura [a,b,*]

[a] Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain
[b] Knowledge Discovery and Intelligent Systems in Biomedicine Laboratory, Maimonides Biomedical Research Institute of Córdoba, Spain
[c] Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284, USA
[d] Polish Academy of Sciences, Institute of Theoretical and Applied Informatics, Gliwice, Poland

ARTICLE INFO

Article history:
Received 29 August 2019
Received in revised form 24 February 2020
Accepted 11 March 2020
Available online 13 March 2020

Keywords:
Multi-label classification
Ensemble
Evolutionary algorithm

ABSTRACT

The multi-label classification task has gained a lot of attention in the last decade thanks to its good application to many real-world problems where each object could be attached to several labels simultaneously. Several approaches based on ensembles for multi-label classification have been proposed in the literature; however, the vast majority are based on randomly selecting the different aspects that make the ensemble diverse and they do not consider the characteristics of the data to build it. In this paper we propose an evolutionary method called Evolutionary AlGorithm for multi-Label Ensemble opTimization, EAGLET, for the selection of simple, accurate and diverse multi-label classifiers to build an ensemble considering the characteristics of the data, such as the relationship among labels and the imbalance degree of the labels. In order to model the relationships among labels, each classifier of the ensemble is focused on a small subset of the label space, resulting in models with a relative low computational complexity and lower imbalance in the output space. The resulting ensemble is generated incrementally given the population of multi-label classifiers, so the member that best fits to the ensemble generated so far, considering both predictive performance and diversity, is selected. The experimental study comparing EAGLET with state-of-the-art methods in multi-label classification over a wide set of sixteen datasets and five evaluation measures, demonstrated that EAGLET significantly outperformed standard MLC methods and obtained better and more consistent results than state-of-the-art multi-label ensembles.

1. Introduction

A large number of problems of classification tasks can be represented as a Multi-Label Classification (MLC) problem, where each of the instances may have several labels associated with them simultaneously [1]. For example, in multimedia annotation or text categorization problems, each could be categorized using several labels or groups simultaneously. Many real-world problems have been successfully solved using this framework, such as protein classification [2], decision support systems for medical diagnosis [3], and image retrieval [4]. The fact of having more than one label associated with each instance, poses new classification challenges that need to be addressed, such as modeling the compound relationships among labels and dealing with the imbalance of the output space. Although imbalance is a difficulty which exists in many other problems, and has been widely studied in the literature [5–7], the problem of dealing with the dependencies among many output labels emerged with MLC. Some studies have already demonstrated that addressing these challenges and dealing with the main characteristics of the multi-labeled data, the predictive results are improved [8–10].

Ensemble-based approaches have been studied and successfully used in many areas of data mining. Ensembles of classifiers are based on the combination of several base classifiers to improve the overall predictive performance; some studies have shown that ensembles outperform single classifiers [11]. Similarly, Ensembles of Multi-Label Classifiers (EMLCs) aim to improve the prediction of simple multi-label classifiers by joining predictions of several multi-label base classifiers. Several approaches were proposed in the literature to build EMLCs, mostly based on building models over different subsets of the training data [12], using different feature spaces [13], or using different subsets of the output space [9]. A thorough description of EMLCs can be found in [14] and [15].

☆ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

* Corresponding author at: Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain.
E-mail address: sventura@uco.es (S. Ventura).

https://doi.org/10.1016/j.knosys.2020.105770
0950-7051/

Despite the fact that EMLCs tend to perform better than single classifiers, the selection of the ensemble members is not a trivial key point [16]. Using accurate classifiers in the ensemble is obviously necessary. However, an ensemble that contains base classifiers that are very similar to each other, although being accurate, not only may not perform as well as expected, but it can even perform worse than individual classifiers. An ensemble containing diverse base classifiers should lead to a better accuracy thanks to the diversity in their outputs, although formal proof of this dependency does not exist [17,18]. Therefore, the selection of base classifiers is key in generating the ensemble. On the other hand, in multi-label scenarios one usually deals with problems of high dimensional output space, so the problems can be intractable with certain algorithmic approaches [19]. As a consequence, selecting a good technique to solve the problem is another key point to be taken into account.

Evolutionary algorithms (EAs) are biology-inspired search algorithms that have been successfully used in different fields of data mining [20–23]. EAs have been also used in multi-label learning tasks such as optimization of base multi-label classifiers [24], and generation of ensembles for both classification and regression problems [25,26]. EAs not only provide a valuable framework to obtain an optimal structure for the EMLC, but also allows to consider the whole characteristics of the data when building the EMLC. Our proposed method, called Evolutionary AlGorithm for multi-Label Ensemble opTimization and hereafter referred as EAGLET, is able to take advantage of the useful tips that the characteristics of the data provide. EAGLET focuses on building an EMLC by selecting simple, accurate and diverse classifiers. Each base classifier is focused on a small subset of labels, considering the relationship among labels and being able to model the compound dependencies among them with a relatively low computational cost. Modeling subsets of labels implies low imbalance in each of the multi-label classifiers, not only making easier the learning phase, but also improving the predictive performance of each model. The imbalance of the data is considered when selecting the members of the ensemble; therefore, EAGLET select accurate but also diverse classifiers in such a way that individuals predicting labels that infrequently appear in the ensemble are more likely to be selected. In this way, EAGLET ensures that all labels are included in the ensemble, while not neglecting infrequent labels.

The experimental study carried out over 16 multi-label datasets and using five evaluation measures demonstrated that EAGLET outperformed other method based on evolutionary algorithms to construct EMLCs [25]. Further, EAGLET outperformed other standard and baseline MLC methods, as well as obtained more consistent performance than state-of-the-art EMLCs, being the only one that did not perform statistically worse than any of the methods.

The rest of the article is organized as follows: Section 2 includes background and related work in MLC, Section 3 presents our proposal of evolutionary algorithm for the combination of accurate and diverse multi-label classifiers into an EMLC, Section 4 shows the experimental setup, Section 5 describes and discusses the results, and finally Section 6 ends with conclusions.

2. Background

2.1. Formal definition of MLC

Let be $\mathcal{D}$ a multi-label dataset composed by a set of m instances, defined as $\mathcal{D} = \{(\mathbf{x}_i, Y_i) | 1 \leq i \leq m\}$. Let $\mathcal{X} = X_1 \times \cdots \times X_d$ be the d-dimensional input space, and $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ the output space composed by $q > 1$ labels. Each multi-label instance is composed by an input vector $\mathbf{x}$ and a set of relevant labels associated with it $Y \subseteq \mathcal{Y}$. Note that each different Y is also called labelset [1].

The goal of MLC is to construct a predictive model $h : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$ which provides a set of relevant labels for an unknown instance. Thus, for each $\mathbf{x} \in \mathcal{X}$, a bipartition $\left(\hat{Y}, \overline{\hat{Y}}\right)$ of the label space $\mathcal{Y}$ is provided, where $\hat{Y} = h(\mathbf{x})$ is the set of relevant labels and $\overline{\hat{Y}}$ the set of irrelevant ones.

Further, lets define an EMLC as a set of n multi-label classifiers. Each of the classifiers h_j provides prediction $\hat{\mathbf{b}}_j = \{\hat{b_{j1}}, \hat{b_{j2}}, \ldots, \hat{b_{jq}}\}$ for all (or part of) the labels, each b_j being 1 if the label is relevant and 0 otherwise. The final prediction of the ensemble is calculated given the average value of the predictions for each label $\hat{\mathbf{v}} = \{\hat{v_1}, \hat{v_2}, \ldots, \hat{v_q}\}$, where the $\hat{v_l}$ for each label λ_l is calculated as $\hat{v_l} = \frac{1}{n}\sum_{j=1}^{n} \hat{b_{jl}}$. Other methods instead of simple voting could be used in order to combine predictions of multi-label classifiers in the ensemble [27].

2.2. MLC algorithms

MLC algorithms are categorized into three main groups: problem transformation, algorithm adaptation, and EMLCs [1].

Problem transformation methods transform a multi-label problem into one or several single-label problems, solving each new problem using traditional single-label algorithms. One of the most popular methods is Binary Relevance (BR) [28] which decomposes the multi-label learning problem into q independent binary classification problems, one for each label. The fact that BR treats each label separately makes it simple, highly parallelizable and resistant to overfitting label combinations, but it does not take into account label combinations so makes it unable to model possible dependencies among the labels. Label Powerset (LP) [29] generates one single-label dataset where each distinct labelset is considered as a different class. LP takes into account label correlations but its complexity is exponential with the number of labels. Also, it is not able to predict a labelset that does not appear in the training dataset and, since many labelsets are usually associated with only few examples, may lead to a highly imbalanced dataset which would make the learning process more difficult and less accurate.

Many other methods were proposed in the literature to overcome the disadvantages of BR and LP. LPBR [30] identifies groups of dependent labels and builds a LP for each of these groups and a BR for each independent label. Pruned Sets (PS) [31] is based on LP, but prunes the instances with infrequent labelsets and then reintroduces them using more frequent subsets of their labels. Classifier Chains (CC) [32] is based on the idea of chaining binary models in such a way that each of them includes as input features the outputs of the previous models in the chain. Genetic Algorithm for ordering Classifier Chains (GACC) [24] uses a genetic algorithm for selecting the most appropriate chain for CC. Each individual in GACC represents a label permutation, i.e., a chain for CC, and they are evaluated using a linear combination of three multi-label evaluation measures. Label specIfic FeaTures for multi-label learning (LIFT) [33] uses clustering of the feature space to select a subset of features that best discriminate each label independently; it also deals with the imbalance in the output space.

Algorithm adaptation methods utilized almost all single-label classification techniques to directly handle multi-label data. Predictive Clustering Trees (PCTs) [34] are decision trees where the data is partitioned at each node by means of a clustering algorithm. These trees are able to deal with multi-label data since the distance between two instances for the clustering algorithm is defined as the sum of Gini Indices of all labels, being able to model the relationship among labels. The well-known

instance-based k-NN method was also adapted to MLC. Multi-Label k-Nearest Neighbors (ML-kNN) [35] deals with multi-label data by finding the k nearest neighbors, counting the number of neighbors belonging to each label, and finally using the *maximum a posteriori* principle to predict the labels for the given instance. Neural networks were also adapted to MLC, such as the Back-Propagation for Multi-Label Learning (BP-MLL) [36], where a new error function taking into account the predicted ranking of labels, and thus considering the relationship among labels was proposed. Multi-Layer Extreme Learning Machine RBF (ML-ELM-RBF) [37] implements a deep network based on radial basis functions for multi-label learning. A thorough description of MLC algorithm adaptation methods can be found in [1].

The third group of methods consists of the EMLCs. Although some methods such as BR or CC combine the predictions of several classifiers, only those that combine the predictions of several multi-label classifiers are considered as EMLCs. RAndom k-labELsets (RAkEL) [9] is one of them with good performance [15]. It breaks the full set of labels into random labelsets of small size, and then trains a LP over each subset of labels of size k (a.k.a. k-labelset). RAkEL is much simpler than LP since it only considers a small subset of labels at once; it takes into account the labels relationship but avoids complexity in the output space that LP might have. The base classifiers of RAkEL include a much more balanced distribution of the classes than using directly LP with all labels. RAkEL overcomes the problem of not being able to predict a labelset that does not appear in the training dataset by means of voting. However, as RAkEL selects its k-labelsets randomly, it cannot guarantee neither that all labels will be used nor the number of times that each label appears in the ensemble. Besides, in selecting the k-labelsets the nature of the data or relationships among labels are not taken into account.

Other EMLCs have been also proposed in the literature. Ensemble of BRs (EBR) [8] combines several BR classifiers built over different subsets of the data. Ensemble of Classifier Chains (ECC) [8] combines several CCs built over different subsets of the data and different random chains. Multi-Label Stacking (MLS) [38] builds q binary models in a first stage, and other q more by extending the input space with label predictions of previous stage. Ensemble of Pruned Sets (EPS) [31] builds an ensemble of PSs, each built over different data subsets. Hierarchy Of Multi-label classifiERs (HOMER) [39] builds a tree-shaped hierarchy of simpler multi-label models, where each node contains a subset of the labels on its parent node. Random Forest of Predictive Clustering Trees (RF-PCT) [40] builds an ensemble of n PCTs by selecting a random subset of the instances in each model, and selecting at each node of the tree the best feature from a random subset of the original ones. Dynamic selection and Circulating Combination-based Clustering (D3C) [41] uses a dynamic ensemble method, where several single-label classifiers of different type are built for each label independently and then uses clustering and dynamic selection to select a subset of accurate and diverse base methods for each of the labels. Ensemble of Multi-Label Sampling (EMLS) [42] builds an ensemble where each member not only is built over a random sample of the data, but also uses a resampling technique, Multi-Label Synthetic Oversampling based on the Local distribution (MLSOL), which aims to deal with the imbalance of the data. A more complete study of state-of-the-art EMLCs can be found in [15].

Finally, an Evolutionary algorithm for building Multi-label Ensembles (EME) was recently proposed [25]. In EME, each individual of the population was a complete EMLC, where each of the base classifiers is centered only in a small k-labelset, similarly to what RAkEL does. In contrast to RAkEL, EME considers the characteristics of the data to build the ensemble, instead of randomly selecting the k-labelsets, which contributed to a better performance than RAkEL and other state-of-the-art EMLCs. However, the fact of evolving the whole ensemble as an individual, made EME computationally complex.

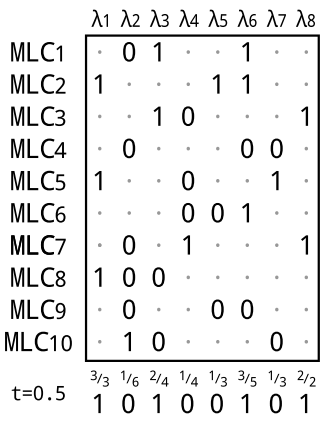


Fig. 1. Prediction of an EMLC for a given instance, using $k = 3$, and prediction threshold $t = 0.5$.

3. EAGLET

In this section, we introduce EAGLET. First, the structure of the multi-label classifier obtained as solution is presented and analyzed. Then, the main aspects of the evolutionary algorithm are presented, including description of the individuals and their initialization, the genetic operators, the fitness function, and how the ensemble is generated from the individuals.

3.1. Ensemble of multi-label classifiers

The aim of EAGLET is to obtain an EMLC, where each of its members is a multi-label base classifier focused only on a small set of labels of size k (k-labelset). Any multi-label base classifier can be used, but we will use LP, as in [9] and [25]. Thus, each classifier is able to model the relationships among labels but with a lower complexity than using the entire output space. As the number of possible combinations of labels in the reduced label space is lower, the imbalance of the output space is consequently reduced in each of the ensemble members.

For an unseen instance, each classifier gives prediction for the labels in its own k-labelset. Then, outputs of all the classifiers are used for the final prediction of the ensemble. For each label, the ratio of positive predictions is calculated, and if it is greater or equal than a given threshold, it results in the final positive (relevant) prediction for this label; otherwise it is irrelevant. Fig. 1 shows an example of how the EMLC combines the predictions for a given instance. The classifier MLC_1 had in its k-labelset the labels λ_2, λ_3, and λ_6, so it can give prediction only for these labels. For instance, for label λ_1, the three classifiers that had it in their k-labelsets give a positive prediction, so, for a threshold $t = 0.5$, the final prediction of the ensemble is positive (relevant). On the other hand, for label λ_2, as only one of the six possible votes were positive, the final prediction is negative (irrelevant).

3.2. Individuals

The individuals of the population in the EA represent each a multi-label base classifier. Each individual is encoded as a binary vector of q elements, where the lth element with a 1 indicates if the label λ_l is included in the multi-label classifier, and with 0 if it is not. Each individual has k elements to 1, k being a parameter of the algorithm. Fig. 2 shows an example of some individuals. For example, for the first of them, which has labels λ_2, λ_3, and λ_6 active, the training dataset is filtered keeping only these labels, and therefore the classifier that it encodes is focused only on their prediction.

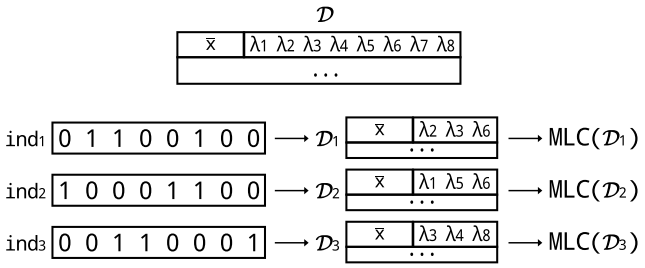


Fig. 2. Example of individuals of the evolutionary algorithm.

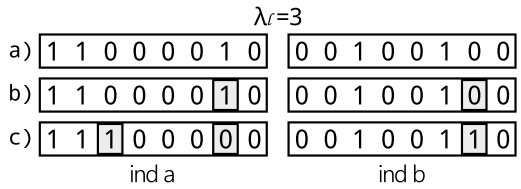


Fig. 3. Fixing the individual creation when a label cannot be shared in a_l individuals.

In contrast to EME, which encodes the entire EMLC into an individual, in EAGLET we encode each base classifier of the ensemble in a different individual. In this way, we better know if each of the ensemble members is positively contributing to the final prediction or not, as they are evaluated separately. In addition, since EME evaluates the entire ensemble, it is not able to determine if any of its members is deteriorating its final performance. The fact of encoding and evaluating each member separately, entails the need of EAGLET to determine how to construct the ensemble; this procedure is analyzed in Section 3.6.

3.3. Initialization

The individuals are created at the beginning of the evolution by using the frequency of each label. The frequency of a label is a representative of its importance or even difficulty of modeling. However, this assumption should be used cautiously as minority labels can be also important. Thus, although the more frequent labels appear more in the initial population, infrequent labels must also be considered and ensured they are present both in the initial population and in the final ensemble. Therefore, the generation of the initial population is biased so that the more frequent labels appear in more individuals than the infrequent ones. At the same time, we ensure that all labels appear on the initial population a minimum number of times, so infrequent labels are taken into account.

The process of creating the initial population is divided into two phases. First, the number of times that each label will appear in the initial population is calculated, and then the individuals are created using the calculated number of appearances.

In order to calculate how many times each label will appear in the initial population, we define as $k \times popSize$ the total number of active bits among all the initial individuals, $popSize$ being the number of individuals in the population. As it has been described above, each label is forced to appear at least a_{min} times in the initial population. Thereby, the remaining $r = k \times popSize - q \times a_{min}$ bits are shared among different labels by using their frequencies. The number of times, a_l, that each label λ_l will appear in the initial population is calculated by Eq. (1), where f_l stands for the frequency of the lth label.

$$a_l = \max\left(popSize, a_{min} + \left\| \frac{f_l}{\sum_{j=1}^{q} f_j} \times r \right\|\right) \tag{1}$$

In this way, the number of times that each label appears in the initial population is given by its relative frequency. Since each label can appear only once in each individual, the number of appearances of each label is upper bounded by the number of individuals of the population. As $k \times popSize$ bits must be shared, if the sum of active bits shared to all labels were less than $k \times popSize$, the remaining bits would be evenly distributed among those labels with fewer number of appearances. This adjustment is performed as follows: (I) labels are sorted upwards by frequency; (II) the number of bits to share for the more infrequent label is incremented in one; (III) until the number of bits to share is adjusted, the number of bits to share for next infrequent label is incremented. A similar process is followed if more than $k \times popSize$ bits are shared, decreasing the number of shared bits to those labels which are more frequent.

Once the number of times that each label appears in the initial population is calculated, the individuals are created. First, $popSize$ empty individuals (all bits are 0) are created. Then the labels are ordered by descending frequency order. For each label λ_l, a_l individuals where λ_l is still not present are randomly selected, activating the lth bit. If for a given label λ_l there are less than a_l possible individuals, this issue would be fixed as shown in Fig. 3. Suppose as example, that when selecting the individuals where λ_3 is to be active, there are less than a_3 available individuals. So first, two individuals are selected randomly: one not including λ_l as active label, and the other having active λ_l and less than k active labels (Fig. 3a). Then, a random active bit from the first individual (that is not active in the second) is selected (Fig. 3b). Finally, label λ_l is activated in the first individual, while the previously selected bit is deactivated in the first individual and activated in the second (Fig. 3c).

At the end of the initialization, individuals that are repeated, if any, are replaced by randomly initialized individuals, created by randomly selecting k different active bits.

3.4. Genetic operators

In this section, both the crossover and the mutation operators are explained. Tournament selection is applied to the population to select the individuals that will form the set of parents. Then, each individual of this set is subject to the crossover and mutation based on their probabilities, p_c and p_m.

3.4.1. Crossover operator

Given two individuals ind_1 and ind_2, the crossover operator swaps information among them in order to obtain individuals including information of both parents. Fig. 4 shows an example of how the crossover operator works. First, the operator identifies the positions of the binary vectors where the individuals differ, creating two sets ds_1 and ds_2 with the positions that are active in one individual but not in the other (Fig. 4a). The elements in each set are shuffled and divided later by the midpoint (Fig. 4b). Then, two new sets $ds_1{}'$ and $ds_2{}'$ are created: $ds_1{}'$ is created from the first half of ds_1 and the second of ds_2, and $ds_2{}'$ is created from the second half of ds_1 and the first of ds_2 (Fig. 4c). Finally, the new individuals $ind_1{}'$ and $ind_2{}'$ are created by copying the bits that matched in ind_1 and ind_2, and activating the bits of $ds_1{}'$ in $ind_1{}'$ and those of $ds_2{}'$ in $ind_2{}'$ (Fig. 4d).

In this way, each new individual inherits information from both parents, which helps to obtain new combination of labels that may perform better. The individuals created by the crossover operator are always valid, since the number of active bits and the structure of each individual is not modified.

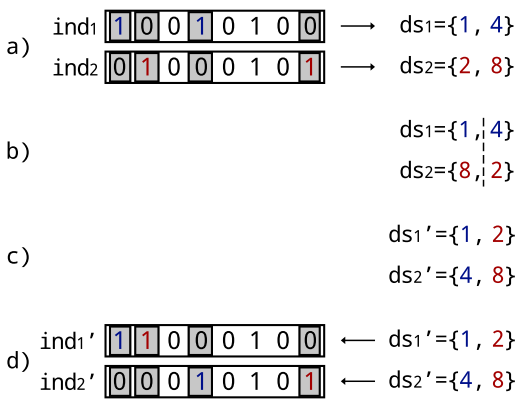


Fig. 4. Example of crossover operator.

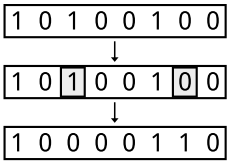


Fig. 5. Example of mutation operator.

3.4.2. Mutation operator

The mutation operator works as shown in Fig. 5: given an individual, two bits with different value are randomly selected and swapped. This stops the individual from taking into account that label but using instead a random one, looking for new combinations of labels. As the number of active labels in the individual is not modified, the generated individuals are always valid.

3.5. Fitness function

The fitness of each individual is calculated using as a measure the performance of the multi-label classifier that it encodes. Many of these evaluation measures are non-decomposable, evaluating the multi-label prediction as a whole [43]. As our approach is based on modeling label dependencies in small subsets of labels, a non-decomposable measure that implicitly considers the relationships among labels is a good choice to evaluate the individuals. The Example-based FMeasure (ExF), presented in Eq. (2), adapts the FMeasure metric to multi-label classification as non-decomposable measure, calculating the FMeasure for each instance, and then averaging [19]. Evaluation measures are indicated as ↑ if they are maximized and with ↓ if they are minimized. FMeasure is a robust evaluation measure which is less sensitive to the imbalance of the labels prediction (in MLC, most predictions tend to be 0, so the output is usually imbalanced). For each individual, the corresponding multi-label classifier is built, and the ExF is calculated over the training set to obtain its fitness. Note that the full training set is used both for training the model and evaluating it.

$$\uparrow \text{ExF} = \frac{1}{m}\sum_{i=1}^{m}\frac{2|\hat{Y}_i \cap Y_i|}{|\hat{Y}_i| \cup |Y_i|} \tag{2}$$

Note that the same individuals could appear in the subsequent populations from one iteration to another. As the evaluation of an individual could be a time-consuming procedure, a table storing the fitness of each individual is created. Therefore, before evaluating an individual, it is searched if it appears in the table, obtaining its fitness and saving the time of evaluating the individual.

3.6. Ensemble generation

The objective of EAGLET is to build an EMLC given the individuals of the population, where each of them is a multi-label classifier. One of the characteristics that make an ensemble perform better than single classifiers is the inclusion of a group of diverse classifiers [18], so the selection of classifiers that will form the ensemble is not straightforward. Therefore, the goal is to generate an ensemble not only taking into account the predictive performance of each of the individuals but also how diverse is the ensemble that the algorithm is generating.

Considering the intrinsic imbalance in multi-label problems, as well as the importance of not neglecting any label when building a model, we consider that all labels should appear a similar number of times in the final ensemble. For this purpose, we define the expected number of appearances or votes in the ensemble for each label as $\frac{n\times k}{q}$, n being the number of classifiers in the ensemble. In this way, EAGLET should not allow some labels to appear a high number of times in the ensemble while others not being present at all.

The process to generate the ensemble, given the population p, is shown in Algorithm 1 and works as follows. At the beginning, the array **eV** storing the expected number of votes for each label is initialized with even values. Then, the best individual according to its fitness is selected and added to the ensemble **e**. Once one individual is added to the ensemble and removed from p, **eV** is updated by decreasing by one vote each of the labels that appear in the added individual, but always ensuring that all positions in **eV** are greater than 0. Then, until the ensemble reaches the desired number of n members, the distance among each individual in p and the current ensemble is calculated. This distance is calculated as a modified Hamming distance, using the array of expected votes as weight vector. The vector of weights **w** to calculate the Hamming distance is obtained by normalizing **eV** in such way that $\sum_{l=1}^{q} w_l = 1$. Then, the Hamming distance between an individual *ind* in p and the ensemble **e** of current size n' is calculated using Eq. (3), where $[\![\pi]\!]$ returns 1 if predicate π is true and 0 otherwise. Thereby, the distance among two individuals is lower if they differ in a label that is more present in the current ensemble than if they differ in a label that appears less frequently, favoring the selection of individuals with labels that rarely appear in the ensemble. The distance to the ensemble is calculated among each of the members of the ensemble (e^i) and then is averaged by the number of members in that moment. Afterwards, the individual that maximizes the linear combination of its fitness and the distance to the ensemble, as $\beta * hd + (1 - \beta) * fitness$, is added to **e** and removed from p. The β value could be modified in order to give more importance to the performance of the individuals or to the distance to the current ensemble. In this way, we generate the ensemble with high performing and accurate individuals, which also results in a diverse ensemble. While $n' < n$, the process selects and add one individual from p to **e** in each iteration.

$$hd_{ind} = \frac{1}{n'}\sum_{i=1}^{n'}\sum_{l=1}^{q}\left(w_l \times [\![ind_l \neq e^i_l]\!]\right) \tag{3}$$

If for any reason, at the end of the ensemble generation any label λ_l was not included in any of the members, it is fixed as follows. First, a random member including one of the labels that is more frequent in the ensemble (but not including any label appearing only once) is removed from **e**. Then, the distance to the ensemble of all the individuals in p containing λ_l is calculated, and from those, which best fits to the ensemble considering the lineal combination of fitness and distance, is included in **e**. In this way, we ensure that the ensemble includes all labels

Algorithm 1 Ensemble generation.

Input p: population of *popSize* individuals.
Output **e**: ensemble of n multi-label classifiers.

eV ← calculate expected votes array.
$b \leftarrow \underset{ind}{\text{argmax}}(fitness_{ind})$
e ← $\{b\}$
$n' \leftarrow 1$
$p \leftarrow p \setminus \{b\}$
eV ← $update$(**eV**, b)
while $n' < n$ **do**
  for each individual *ind* in p **do**
    $hd_{ind} \leftarrow HammingDistance(ind, \mathbf{e}, \mathbf{eV})$
  end for
  $b \leftarrow \underset{ind}{\text{argmax}}(\beta * hd_{ind} + (1 - \beta) * fitness_{ind})$
  e ← **e** $\cup \{b\}$
  $n' \leftarrow n' + 1$
  $p \leftarrow p \setminus \{b\}$
  eV ← $update$(**eV**, b)
end while
return **e**

Since the ensemble generated with the final population is not ensured to be the best, an ensemble is created and evaluated in each generation using ExF measure; if it is better than the best found so far, it is stored as the new best.

Further, this process of selecting the individuals is also useful for updating the population from one generation to another. In this way, p_g being the population in the generation g, and s_g the crossed and mutated individuals in generation g; p_{g+1} is created as follows: (I) p_g and s_g are combined into c_g; (II) the individuals which are repeated in c_g are removed; (III) the n individuals selected to form the ensemble are copied to p_{g+1} and removed from c_g; and (IV) $popSize - n$ individuals are randomly selected from c_g and copied to p_{g+1}, where individuals with greater fitness have higher probability to be selected. Thus, we pass to the next generation not only high performing but also diverse individuals.

3.7. Time complexity

As in most EAs, the part of EAGLET that is most time-consuming is the evaluation of the individuals. To evaluate each individual, the multi-label classifier that it encodes is built and then evaluated over the training dataset. The total number of individuals that EAGLET evaluates is upper bounded by $popSize \times G$, being G the number of generations; in some cases, the number of possible different individuals, i.e., possible combinations of activating k bits into a set of q labels, is lower than $popSize \times G$, so the maximum number of individuals to evaluate is $\binom{q}{k}$. Note that EAGLET includes a table where the fitness of all evaluated individuals is stored, so in practice, the number of individuals to evaluate is drastically reduced. Further, the evaluation of individuals in EAGLET is performed in parallel, so as many individuals as available threads can be evaluated in parallel.

We decided to use C4.5 as single-label classifier in EAGLET, which complexity is $\mathcal{O}(m \times d^2)$, m being the number of instances and d the number of features of the dataset [44]. If other base classifier were used, the time complexity of EAGLET would change in the same way as the single-label classifier used, but we analyze the complexity of EAGLET according to C4.5. Therefore, the time complexity of EAGLET is $\mathcal{O}(m \times d^2 \times n_T)$, being n_T the total number of individuals evaluated, as $n_T = \min(popSize \times G, \binom{q}{k})$.

Table 1
Datasets and their characteristics. The datasets are ordered by the number of labels.

Dataset	m	d	q	$card$	$avgIR$	$rDep$	$comp$
Emotions	593	72	6	1.868	1.478	0.933	2.56E5
Reuters1000	294	1000	6	1.126	1.789	0.667	1.76E6
Guardian1000	302	1000	6	1.126	1.773	0.667	1.81E6
Bbc1000	352	1000	6	1.125	1.718	0.733	2.11E6
3s-inter3000	169	3000	6	1.142	1.766	0.400	3.04E6
Gnegative[a]	1392	1717	8	1.046	18.448	0.536	4.90E6
Plant[a]	978	440	12	1.079	6.690	0.318	5.16E6
Water-quality	1060	16	14	5.073	1.767	0.473	2.37E5
Yeast	2417	103	14	4.237	7.197	0.670	3.49E6
Human[a]	3106	440	14	1.185	15.289	0.418	1.91E7
Birds	645	260	19	1.014	5.407	0.123	3.19E6
Genbase	662	1186	27	1.252	37.315	0.157	2.12E7
Medical	978	1449	45	1.245	89.501	0.039	6.38E7
NusWide[b]	2696	128	81	1.863	89.130	0.087	2.80E7
Stackex coffee	225	1763	123	1.987	27.241	0.017	4.88E7
CAL500	502	68	174	26.044	20.578	0.192	5.94E6

[a]These datasets correspond to the PseAAC version of dataset available in the repository.
[b]A random selection of the original instances of *NusWide cVLAD+* dataset was performed in order to be able to execute it in a reasonable time.

4. Experimental studies

In this section, the datasets and evaluation measures used for assessing the algorithms are described, and the experimental settings are explained.

4.1. Datasets

A set of 16 multi-label datasets was selected from the data repository at http://www.uco.es/kdis/mllresources/. The selected datasets cover a wide range of labels from 6 to 174. For the larger dataset, there are almost one million of possible combinations of k-labelsets with $k = 3$. The number of possible combinations of these individuals into an ensemble is much higher. The immense variety of different possible k-labelsets stresses the importance of having a method which optimizes the combination of labels and selection of multi-label classifiers for the ensemble according to its performance. Table 1 shows the datasets along with their characteristics, such as number of instances (m), number of attributes (d), number of labels (q), cardinality ($card$), average imbalance ratio ($avgIR$), ratio of dependent label pairs ($rDep$), and complexity ($comp$). The $avgIR$ measures the imbalance of the labels, the greater the value, the greater the imbalance of the dataset; $rDep$ measures the relationship among pairs of labels, the greater the value, the greater the conditional dependence among labels in the dataset; $comp$ is defined as $m \times d \times q$, the greater the value, the higher the complexity of the dataset. These datasets were selected according to a wide range of $card$, $avgIR$, and $rDep$ values to have a diverse set of multi-label datasets with different characteristics. The characterization and partitions of the datasets was performed using MLDA tool [45].

4.2. Evaluation measures

For assessing different multi-label classifiers in the experiments, several evaluation measures are used [28]. Hamming loss (HL) is one of the most used evaluation measures in multi-label classification. HL measures, on average, how many labels have been correctly predicted, taking into account both prediction and omission errors, i.e., where a negative label is predicted or a positive label is omitted, respectively. HL is a measure to be minimized as defined in (4), Δ being the symmetric difference between two binary sets. HL tends to be 0 in datasets where

the number of labels is large but only a small subset of them appears in each instance. So using other evaluation measures is recommended to have a better assessment of performance of the classifiers [46]. Subset Accuracy (SA) is a strict but often used measure in MLC, and it is defined in Eq. (5). It evaluates the ratio of full correctly predicted instances, i.e., the true and predicted labels for a given instance perfect match, including both relevant and irrelevant labels.

$$\downarrow \text{HL} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{q}|Y_i \Delta \hat{Y}_i| \tag{4}$$

$$\uparrow \text{SA} = \frac{1}{m}\sum_{i=1}^{m}[\![Y_i = \hat{Y}_i]\!] \tag{5}$$

On the other hand, classic evaluation measures for binary classification have been extended to MLC scenarios, such as *example-based*, *micro-averaged*, and *macro-averaged* approaches [1]. We used *macro-averaged* evaluation measures to compare different MLC state-of-the-art methods. They give the same importance to all labels in their calculation, so infrequent labels are taken into account and not neglected to calculate the value of the measure. Precision (MaP), recall (MaR), and specificity (MaS) are defined in their *macro-averaged* approach in Eqs. (6)–(8) respectively, tp_i, tn_i, fp_i, and fn_i being the true positives, true negatives, false positives, and false negatives respectively for the ith label. With these measures we evaluate both the ratio of correctly predicted relevant labels (MaP and MaR), as well as the ratio of correctly predicted irrelevant labels (MaS). These 5 evaluation measures are sufficient to assess the performance of MLC methods [47].

$$\uparrow \text{MaP} = \frac{1}{q}\sum_{i=1}^{q}\frac{tp_i}{tp_i + fp_i} \tag{6}$$

$$\uparrow \text{MaR} = \frac{1}{q}\sum_{i=1}^{q}\frac{tp_i}{tp_i + fn_i} \tag{7}$$

$$\uparrow \text{MaS} = \frac{1}{q}\sum_{i=1}^{q}\frac{tn_i}{tn_i + fp_i} \tag{8}$$

4.3. Experimental settings

The goal of the experimental study is to compare the performance of EAGLET with other state-of-the-art MLC methods. For that, first previous experiments to select some of the parameters of EAGLET were carried out. Then, given that both EAGLET and EME present some similarities, their performance was compared. Subsequently, EAGLET was compared with standard MLC methods such as BR, LP, CC, GACC, PS, LPBR, and LIFT; and also with other state-of-the-art EMLCs such as ECC, EBR, RAkEL, EPS, HOMER, MLS, RF-PCT, D3C, and EMLS. The runtime of EAGLET was analyzed and compared to other MLC methods.

For the experiments, the datasets were partitioned using random 5-fold cross-validation procedure. Since many multi-label datasets are high imbalanced in the output space, with many labels rarely appearing, using a higher number of partitions may result in the loss of some labels at some partitions. Each method was executed over different partitions, and those methods that use random numbers (such as EAGLET, EME, CC, GACC, EBR, ECC, EPS, RAkEL, RF-PCT, and EMLS) were executed using 10 different random seeds over each partition. The evaluation measures were calculated in the test set in each of the partitions, and then the results were averaged. The experiments were performed on a machine with Rocks cluster O.S., Intel Xeon E5645 Processor (6x 2.40 GHz) and 24 GB DDR4 RAM.

For all methods, the default parameters proposed by original authors were used. EME was executed using 50 individuals in all cases, while the number of generations varied from 110 to 300 depending on the dimensionality of the dataset. The probability to mutate was set to $p_m = 0.2$, and the probability of crossover was set to $p_c = 0.9$ for datasets with $q \leq 8$ and to $p_c = 0.8$ for those datasets with $q > 8$. Further, the EMLC built by EME used $n = 2q$ classifiers with subsets of $k = 3$ labels. GACC was executed using 20 generations and 35 individuals for datasets with complexity lower than 1E07, and with 15 generations and 20 individuals for those with higher complexity. LIFT fixed the ratio for controlling the number of clusters to $r = 0.1$. Both PS and EPS pruned the instances with labelsets occurring less than 3 times, and kept the top two best ranked subsets when reintroducing the pruned instances.[1] EPS included $n = 10$ members in the ensemble, each of them built over a sample without replacement of the original training data. Both EBR and ECC used $n = 10$ and sampling with replacement of the original training set. RAkEL used $n = 2q$ classifiers with subsets of $k = 3$ labels. HOMER generated $c = 3$ clusters at each node and used the *balanced k-means* clustering method. RF-PCT used 10 trees in the ensemble, each using the full training set. EMLS used a sampling ratio $r = 0.3$ for the MLSOL resampling method, and $n = 10$.

EAGLET, whose source code is publicly available in a GitHub repository,[2] was implemented using the JCLEC [48] and Mulan [49] frameworks. We fixed EAGLET to use k-labelsets of size $k = 3$ in each of the members of the ensemble [9]. In order to have, on average, 10 votes for each label (as in state-of-the-art EMLCs such as EBR, ECC, EPS, RF-PCT, or EMLS), the ensemble was composed by $n = \|3.33q\|$ members. In the individual initialization, we set $a_{min} = 1$ in order to ensure that all labels are considered in the initial population. Finally, the selection of most of its parameters, such as the value of β in the ensemble generation, the number of individuals, number of generations, and crossover and mutation probabilities, is described in Section 5.1.

All methods used C4.5 (Weka's J48 [50]) as a single-label classifier given its generalized use in EMLCs [8,9,51]. The minimum number of objects per leaf was set to 2, and the pruning confidence to 0.25. We used the same parameters for C4.5 for all methods to compare the MLC approaches under the same conditions; by optimizing the parameters of C4.5, the performance of all MLC methods could be improved, but the optimization of these parameters is not in the scope of this paper.

Finally, for the comparison of the predictive performance of different methods, the Skillings–Mack's test [52] was performed for each evaluation measure. Skillings–Mack's test is similar to the Friedman's test, but is able to deal with missing values (e.g., algorithms that did not finish its execution are considered as missing values). In cases where the Skillings–Mack's test determined that there were significant differences in the performance of the algorithms, the post-hoc Holm's test [53] for multiple comparisons with a control algorithm was also performed. In cases where only two methods were compared, Wilcoxon's signed-rank [54] test was done. In order to perform multiple comparisons without a significant level, and therefore providing more statistical information, the adjusted p-values were used [55].

5. Results and discussion

In this section, a summary of the results of the different experiments defined in previous section is presented. The supplementary material available at the KDIS Research Group website contains full tables with results, including not only the five evaluation measures presented in the paper, but also many more.[3]

[1] CAL500 dataset has as many different labelsets as instances, so only for it, both PS and EPS were run without pruning the infrequent labelsets.

[2] https://github.com/kdis-lab/EAGLET.

[3] All results are available at http://www.uco.es/kdis/eaglet/.

Table 2
Results of EAGLET and EME for 5 evaluation measures. Best results for each dataset and measure are marked in bold.

	↓HL		↑SA		↑MaP		↑MaR		↑MaS	
	EAGLET	EME	EAGLET	EME	EAGLET	EME	EAGLET	EME	EAGLET	EME
Emotions	**0.204**	0.220	**0.290**	0.248	**0.681**	0.657	**0.638**	0.592	**0.859**	0.858
Reuters1000	**0.227**	0.229	**0.123**	0.111	**0.253**	0.235	**0.141**	0.131	0.912	**0.914**
Guardian1000	**0.227**	0.228	0.083	**0.086**	0.259	**0.284**	**0.151**	0.133	0.917	**0.918**
Bbc1000	**0.211**	0.216	**0.145**	0.120	**0.375**	0.362	0.157	**0.164**	**0.928**	0.922
3s-inter3000	**0.261**	0.265	**0.041**	0.040	**0.120**	0.107	0.069	**0.078**	**0.891**	0.883
Gnegative	**0.084**	0.091	**0.523**	0.487	**0.563**	0.509	**0.377**	0.352	**0.963**	0.961
Plant	**0.099**	0.102	**0.113**	**0.113**	**0.202**	0.183	0.081	**0.082**	**0.974**	0.968
Water-quality	**0.294**	0.299	**0.016**	0.014	**0.559**	0.558	**0.503**	0.469	0.773	**0.786**
Yeast	**0.208**	0.210	**0.151**	0.137	**0.512**	0.510	**0.384**	0.361	0.788	**0.803**
Human	**0.087**	0.090	**0.163**	0.159	**0.241**	0.220	0.092	**0.095**	**0.974**	0.968
Birds	**0.046**	0.047	**0.500**	0.496	**0.422**	0.396	**0.247**	0.228	**0.989**	**0.989**
Genbase	**0.001**	**0.001**	**0.976**	0.966	**0.930**	0.929	**0.938**	0.934	**1.000**	**1.000**
Medical	**0.010**	**0.010**	**0.658**	0.649	0.637	**0.651**	0.640	**0.650**	**0.995**	**0.995**

First, the effect of the parameters of EAGLET was analyzed and their default values were selected. Then, EAGLET was compared with EME, standard MLC methods, and state-of-the-art EMLCs. The runtime of EAGLET and other MLC methods is also analyzed.

5.1. Analysis of the parameters of EAGLET

The performance of EAGLET, as most EAs, depends on several parameters, such as the number of individuals, number of generations, or probabilities of crossover and mutation operators. In order to select the best default parameters for EAGLET, we conducted several experiments over four datasets. These datasets are different from those used in the experimental setup, to not bias further comparisons. PlantGO, Tmc2007-500, Enron, and Mediamill datasets, with 12, 22, 53, and 101 labels respectively, were used. For both Tmc2007-500 and Mediamill datasets, given its large number of instances, a random subset of 10% and 5% of the instances was selected, respectively.

First, EAGLET was executed using different values for the number of individuals and the value of β used in the ensemble generation (see Section 3.6). Note that the lower the value of β, the greater the importance of the fitness of each individual in the member selection for the ensemble. Higher β values mean that we prefer selecting more diverse individuals for the ensemble. The number of individuals was set to $popSize = \{2n, 3n, 4n\}$, n being the number of members in the ensemble. Values of $\beta = \{0.25, 0.5, 0.75\}$ were used. In these experiments, the number of generations was set to 50 in all cases.

In all cases, EAGLET performed better if $\beta = 0.75$ was used; i.e., it built better ensemble methods when more diverse individuals were included. As for the number of individuals, EAGLET performed better with $3n$ individuals in Mediamill, while in the rest of cases, better results were obtained with $2n$ individuals. As the number of labels increases, the number of different possible individuals and also different combinations of individuals in the ensemble grows exponentially; that is why in Mediamill a greater number of individuals is needed to obtain better results. Therefore, given these results, we set $\beta = 0.75$, and $popSize = 2n$ for datasets with less than 100 labels, and $popSize = 3n$ for datasets where the number of labels is greater or equal than 100.

Once we fixed the number of individuals in the population, we performed experiments to select the number of generations. We executed EAGLET over all four datasets using $n_g = 100$ generations. Results show that in generation 50, the fitness value of the best ensemble was stabilized. Therefore, for all further experiments we set the number of generations to 50, since EAGLET obtained good results and reduced the total runtime.

Experiments to select probabilities for both crossover and mutation operators were performed, and values of $p_c = \{0.7, 0.8, 0.9\}$ and $p_m = \{0.1, 0.2, 0.3\}$ were used. For smaller datasets (PlantGO and Tmc2007-500) EAGLET performed better on average using $p_c = 0.7$ and $p_m = 0.2$, so we set these parameters as default for datasets with less than 30 labels. For datasets with larger number of labels, such as Enron and Mediamill, better results were obtained on average using $p_c = 0.7$ and $p_m = 0.1$, so we used them in the remaining experiments conducted over datasets with a large number of labels.

Table 3
Results of the Wilcoxon's test comparing EAGLET and EME. Those p-values lower than 0.05 are marked in bold, indicating that EAGLET performed better than EME at 95% confidence.

	HL	SA	MaP	MaR	MaS
R+	89.5	75.0	74.0	72.0	39.0
R−	1.5	3.0	17.0	19.0	39.0
p-value	**0.0014**	**0.0038**	**0.0409**	0.0592	≥0.2

5.2. Comparing EAGLET and EME

EME is a method based on EAs able to build EMLCs, whose behavior is similar to the EMLC obtained by EAGLET. However, the way in which EAGLET obtains the EMLC is different from EME's. Therefore, comparing the performance of EAGLET and EME is necessary. To make a fair comparison, the results are presented using only 13 of the 16 datasets included in Section 4.1, since EME did not finish its execution on the three datasets with a higher number of labels, due to memory overhead. Results for all evaluation measures and 13 datasets for both EAGLET and EME are shown in Table 2. The best results for each measure and dataset are shown in bold typeface.

It can be observed that for HL, SA, and MaP measures EAGLET obtained better performance than EME in most cases. In MaR and MaS, EME obtained better results; however, EAGLET was still better than EME on most datasets. Given these results, we next performed the Wilcoxon's signed-rank test for each evaluation measure, and the results are shown in Table 3. The test indicates that EAGLET performed statistically better than EME in three of the five evaluation measures with 95% confidence. Therefore, we prove that the fact of evolving the members of the ensemble separately instead of the entire ensemble, lead EAGLET to obtain an EMLC which performs significantly better than EME.

5.3. Comparing EAGLET with standard MLC methods

In this section, we present and discuss the results of the experimental study comparing EAGLET with standard and baseline MLC methods such as BR, LP, CC, GACC, PS, LPBR, and LIFT. Tables 4–8 show the results for HL, SA, MaP, MaR, and MaS measures.

Table 4
Results of ↓HL for EAGLET and standard MLC methods. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	BR	LP	CC	GACC	PS	LPBR	LIFT
Emotions	**0.204**	0.254	0.263	0.262	0.260	0.273	0.252	0.250
Reuters1000	0.227	0.257	0.268	0.284	0.285	0.276	0.257	**0.191**
Guardian1000	0.227	0.265	0.279	0.287	0.289	0.274	0.265	**0.190**
Bbc1000	0.211	0.263	0.264	0.284	0.282	0.270	0.267	**0.187**
3s-inter3000	0.261	0.308	0.312	0.311	0.310	0.314	0.308	**0.190**
Gnegative	0.084	0.120	0.119	0.122	0.121	0.118	0.123	**0.066**
Plant	0.099	0.139	0.141	0.141	0.139	0.144	0.139	**0.084**
Water-quality	0.294	0.310	0.375	0.334	0.334	0.337	0.315	**0.284**
Yeast	0.208	0.249	0.283	0.268	0.265	0.279	0.274	**0.193**
Human	0.087	0.121	0.126	0.122	0.122	0.123	0.121	**0.078**
Birds	**0.046**	0.052	0.063	0.052	0.053	0.054	0.052	DNF
Genbase	**0.001**	**0.001**	0.002	**0.001**	**0.001**	0.004	**0.001**	0.044
Medical	**0.010**	0.011	0.013	**0.010**	**0.010**	0.013	**0.010**	0.012
NusWide	0.024	0.031	0.039	0.030	0.029	0.031	0.033	**0.021**
Stackex coffee	0.016	0.017	0.027	0.017	0.017	0.022	0.017	**0.015**
CAL500	0.154	0.163	0.198	0.175	0.174	0.198	0.189	**0.138**

Table 5
Results of ↑SA for EAGLET and standard MLC methods. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	BR	LP	CC	GACC	PS	LPBR	LIFT
Emotions	**0.290**	0.170	0.226	0.218	0.217	0.209	0.191	0.140
Reuters1000	0.123	0.092	**0.207**	0.163	0.171	0.204	0.092	0.003
Guardian1000	0.083	0.069	0.166	0.147	0.155	**0.192**	0.069	0.000
Bbc1000	0.145	0.071	**0.207**	0.175	0.181	0.202	0.071	0.009
3s-inter3000	0.041	0.089	0.094	0.105	0.105	**0.106**	0.089	0.018
Gnegative	0.523	0.397	0.522	0.503	0.508	0.520	0.422	**0.595**
Plant	0.113	0.099	0.189	0.188	**0.206**	0.172	0.101	0.156
Water-quality	0.016	0.008	0.005	0.010	0.010	0.015	0.008	**0.019**
Yeast	0.151	0.070	0.135	0.138	0.143	0.131	0.113	**0.173**
Human	0.163	0.115	0.175	0.192	**0.194**	0.187	0.122	0.181
Birds	**0.500**	0.471	0.429	0.476	0.468	0.462	0.471	DNF
Genbase	**0.976**	0.965	0.965	0.965	0.965	0.937	0.965	0.083
Medical	0.658	0.635	0.661	0.664	**0.668**	0.666	0.665	0.598
NusWide	0.188	0.097	0.084	0.143	0.147	0.095	0.092	**0.230**
Stackex coffee	**0.085**	0.067	0.031	0.053	0.056	0.018	0.067	0.053
CAL500	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**

Table 6
Results of ↑MaP for EAGLET and standard MLC methods. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	BR	LP	CC	GACC	PS	LPBR	LIFT
Emotions	**0.681**	0.596	0.569	0.578	0.583	0.560	0.593	0.538
Reuters1000	0.253	0.215	**0.287**	0.211	0.217	0.256	0.215	0.033
Guardian1000	**0.259**	0.205	0.221	0.230	0.237	0.248	0.205	0.000
Bbc1000	**0.375**	0.262	0.293	0.244	0.241	0.291	0.256	0.083
3s-inter3000	0.120	0.167	0.174	**0.177**	0.175	0.154	0.167	0.120
Gnegative	**0.563**	0.317	0.366	0.316	0.323	0.335	0.300	0.557
Plant	**0.202**	0.142	0.144	0.142	0.149	0.123	0.143	0.168
Water-quality	0.559	0.521	0.446	0.500	0.500	0.354	0.510	**0.652**
Yeast	0.512	0.403	0.377	0.394	0.393	0.377	0.381	**0.545**
Human	**0.241**	0.163	0.129	0.143	0.145	0.132	0.158	0.240
Birds	**0.422**	0.398	0.318	0.386	0.369	0.318	0.398	DNF
Genbase	**0.930**	0.929	0.915	0.929	0.714	0.760	0.929	0.245
Medical	0.637	0.644	0.615	**0.646**	0.361	0.621	0.643	0.614
NusWide	**0.171**	**0.171**	0.098	0.165	0.039	0.151	0.159	0.170
Stackex coffee	**0.641**	0.635	0.495	0.635	0.055	0.607	0.635	**0.641**
CAL500	**0.190**	0.168	0.164	0.178	0.149	0.164	0.164	0.105

In both HL and MaS, LIFT was the algorithm that performed the better (in 12 datasets for HL and 13 in MaS), while EAGLET was the best in four and two datasets, respectively, including some ties. For SA, the results were more spread among the different methods. Note that in CAL500 dataset all methods obtained a subset accuracy of 0.0. This dataset has as many distinct labelsets as the number of instances, i.e., each instance has a different subset of labels associated with it.

On the other hand, for MaP, EAGLET obtained the best performance in 11 datasets, followed by LIFT being the best in 3. Further, and although MaP and MaR are opposite measures, i.e., obtaining good results by one usually leads to bad results by the other, EAGLET also obtained a great performance in MaR, only surpassed by LP. This behavior shows consistent performance of EAGLET when compared with standard MLC methods.

In order to determine if there were significant differences between the methods, Skillings–Mack's test was performed. In Table 9, the rankings of each algorithm as calculated by the Skillings–Mack's test are shown. For each dataset and measure, the Skillings–Mack's test gives to the method with better performance a value of ranking of 1, to the second best a ranking of 2, and so on. In cases where the algorithm did not finish its

Table 7
Results of ↑MaR for EAGLET and standard MLC methods. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	BR	LP	CC	GACC	PS	LPBR	LIFT
Emotions	**0.638**	0.547	0.561	0.568	0.573	0.553	0.571	0.349
Reuters1000	0.141	0.178	**0.275**	0.201	0.205	0.235	0.178	0.004
Guardian1000	0.151	0.132	**0.223**	0.196	0.197	0.217	0.132	0.000
Bbc1000	0.157	0.136	**0.291**	0.202	0.206	0.263	0.129	0.008
3s-inter3000	0.069	0.173	**0.206**	0.191	0.186	0.187	0.173	0.023
Gnegative	0.377	0.343	0.368	0.340	0.338	0.327	0.330	**0.385**
Plant	0.081	0.151	0.137	0.151	**0.157**	0.125	0.151	0.076
Water-quality	**0.503**	0.429	0.451	0.445	0.452	0.179	0.440	0.369
Yeast	0.384	0.384	0.375	0.387	**0.395**	0.362	0.388	0.345
Human	0.092	**0.162**	0.130	0.148	0.148	0.134	0.155	0.080
Birds	0.247	0.269	**0.292**	0.264	0.246	0.226	0.269	DNF
Genbase	**0.938**	0.934	0.902	0.934	0.719	0.751	0.934	0.227
Medical	0.640	0.644	0.605	**0.645**	0.361	0.600	**0.645**	0.549
NusWide	0.139	**0.161**	0.099	0.153	0.025	0.142	0.154	0.137
Stackex coffee	**0.636**	0.628	0.498	0.628	0.047	0.613	0.628	0.618
CAL500	0.139	0.146	0.162	0.158	0.129	0.162	**0.163**	0.087

Table 8
Results of ↑MaS for EAGLET and standard MLC methods. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	BR	LP	CC	GACC	PS	LPBR	LIFT
Emotions	0.859	0.829	0.811	0.808	0.810	0.800	0.820	**0.910**
Reuters1000	0.912	0.865	0.833	0.822	0.821	0.834	0.865	**0.995**
Guardian1000	0.917	0.864	0.826	0.823	0.820	0.833	0.864	**0.997**
Bbc1000	0.928	0.864	0.831	0.824	0.825	0.833	0.862	**0.999**
3s-inter3000	0.891	0.801	0.799	0.795	0.797	0.801	0.801	**0.993**
Gnegative	0.963	0.922	0.922	0.919	0.920	0.923	0.917	**0.979**
Plant	0.974	0.916	0.916	0.914	0.914	0.915	0.916	**0.990**
Water-quality	0.773	0.782	0.687	0.750	0.746	**0.899**	0.770	0.845
Yeast	0.788	0.745	0.735	0.743	0.733	0.743	0.730	**0.809**
Human	0.974	0.924	0.923	0.924	0.924	0.925	0.925	**0.987**
Birds	**0.989**	0.982	0.968	0.982	0.981	0.982	0.982	DNF
Genbase	**1.000**	**1.000**	0.999	**1.000**	**1.000**	0.999	**1.000**	0.999
Medical	0.995	0.995	0.993	0.995	0.995	0.994	0.995	**0.997**
NusWide	0.995	0.986	0.979	0.987	0.988	0.987	0.984	**0.998**
Stackex coffee	0.996	0.996	0.986	0.997	0.996	0.992	0.996	**0.999**
CAL500	0.913	0.901	0.856	0.883	0.884	0.856	0.865	**0.949**

Table 9
Average rankings of the Skillings–Mack's test comparing EAGLET with standard MLC methods. Last column shows the average ranking among all measures.

	HL	SA	MaP	MaR	MaS	avgRank
EAGLET	**1.97**	3.53	**1.91**	4.34	2.41	**2.83**
BR	3.75	6.06	3.75	4.16	3.84	4.31
LP	6.78	4.34	5.25	3.72	6.63	5.34
CC	5.34	3.81	4.41	**3.47**	5.63	4.53
GACC	5.06	**3.28**	5.47	4.59	5.94	4.87
PS	6.66	4.09	5.63	5.09	5.09	5.31
LPBR	4.38	5.63	4.63	3.72	4.72	4.61
LIFT	2.06	5.25	4.97	6.91	**1.75**	4.19

Table 10
Results of Holm's post-hoc test comparing EAGLET and standard MLC methods. Algorithms marked with "–" are the control algorithm in each measure, while *p*-values in bold typeface indicate that there are significant differences with the control algorithm at 95% confidence.

	HL	SA	MaP	MaR	MaS
EAGLET	–	≥0.2	–	≥0.2	≥0.2
BR	0.0794	**0.0092**	**0.0333**	≥0.2	**0.0312**
LP	**0.0000**	≥0.2	**0.0006**	≥0.2	**0.0000**
CC	**0.0005**	≥0.2	**0.0078**	–	**0.0000**
GACC	**0.0014**	–	**0.0002**	≥0.2	**0.0000**
PS	**0.0000**	≥0.2	**0.0001**	≥0.2	**0.0005**
LPBR	**0.0164**	**0.0408**	**0.0051**	≥0.2	**0.0018**
LIFT	≥0.2	0.1150	**0.0016**	**0.0005**	–

execution, the average value of ranking for this measure and dataset among the rest of methods is assigned. EAGLET was the control algorithm in two cases, while CC, GACC, and LIFT were in one measure each. We also observe that LIFT, which seemed to have a very great performance in some measures such as HL and MaS, has a poor performance in other measures, including that it has the worst ranking for MaR. For HL, although LIFT was the best method in most datasets, the consistency of EAGLET (which performed the best or second in almost all cases for HL) made it the one with better ranking, followed by LIFT. Further, EAGLET has the better average ranking value among all measures, again demonstrating its consistency. Skillings–Mack's test rejected the null hypothesis that performance of all methods was statistically the same for all measures, with *p*-values 3.02E-11 (HL), 6.47E-3 (SA), 2.74E-4 (MaP), 1.79E-3 (MaR), and 1.49E-9 (MaS).

Then, we performed Holm's test and the results are shown in Table 10. We observe that EAGLET is the only algorithm whose performance is statistically the same than the control algorithm in all cases, at 95% confidence. Let us emphasize in MaP, where EAGLET is the control algorithm and its performance is significantly better than all the rest of methods. Also, in MaS, where LIFT is the algorithm with best performance, EAGLET is the only one whose performance is not significantly worse than LIFT. Therefore, we demonstrated that EAGLET obtained a significantly better performance than standard MLC methods.

5.4. Comparing EAGLET with state-of-the-art EMLCs

In this part of the experimental study, we discuss the experimental results comparing EAGLET with other state-of-the-art

Table 11
Results of ↓HL for EAGLET and state-of-the-art EMLCs. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT	D3C	EMLS
Emotions	0.204	0.200	0.202	0.225	0.209	0.253	0.259	0.213	**0.182**	0.247
Reuters1000	0.227	0.227	0.214	0.237	0.231	0.317	0.256	0.205	**0.198**	0.314
Guardian1000	0.227	0.225	0.210	0.239	0.226	0.288	0.264	0.202	**0.193**	0.319
Bbc1000	0.211	0.216	0.202	0.222	0.221	0.286	0.257	0.197	**0.194**	0.304
3s-inter3000	0.261	0.246	0.220	0.279	0.243	0.302	0.315	0.207	**0.200**	0.362
Gnegative	0.084	**0.082**	**0.082**	0.094	0.086	0.117	0.117	0.092	0.090	0.136
Plant	0.099	0.097	**0.093**	0.107	0.095	0.140	0.137	0.096	0.112	0.192
Water-quality	0.294	0.295	0.290	0.311	0.324	0.341	0.337	0.314	**0.270**	0.304
Yeast	0.208	0.210	0.207	0.225	0.210	0.263	0.273	0.219	**0.204**	0.274
Human	0.087	0.088	**0.085**	0.097	0.087	0.121	0.118	0.090	0.113	0.166
Birds	0.046	**0.043**	**0.043**	0.048	0.046	0.062	0.049	0.046	0.097	0.065
Genbase	**0.001**	**0.001**	**0.001**	**0.001**	0.004	**0.001**	**0.001**	0.046	0.003	0.127
Medical	**0.010**	**0.010**	0.011	0.011	0.012	0.011	0.011	0.025	0.032	0.278
NusWide	0.024	**0.023**	**0.023**	0.026	**0.023**	0.036	0.030	0.024	0.052	0.439
Stackex coffee	**0.016**	**0.016**	**0.016**	**0.016**	0.020	DNF	0.017	**0.016**	0.024	0.608
CAL500	0.154	**0.148**	0.149	0.166	DNF	0.210	0.170	DNF	0.200	0.359

Table 12
Results of ↑SA for EAGLET and state-of-the-art EMLCs. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT	D3C	EMLS
Emotions	0.290	0.297	0.274	0.250	0.292	0.182	0.186	0.284	**0.315**	0.195
Reuters1000	0.123	0.064	0.040	**0.129**	0.115	0.078	0.112	0.045	0.031	0.068
Guardian1000	0.083	0.063	0.037	0.092	**0.130**	0.069	0.076	0.037	0.023	0.075
Bbc1000	**0.145**	0.086	0.057	0.134	0.142	0.102	0.088	0.045	0.043	0.086
3s-inter3000	0.041	0.050	0.025	0.037	0.044	0.042	**0.077**	0.033	0.006	0.041
Gnegative	0.523	**0.548**	0.497	0.493	0.513	0.421	0.397	0.470	0.530	0.386
Plant	0.113	**0.140**	0.089	0.127	0.095	0.094	0.109	0.101	0.118	0.073
Water-quality	0.016	0.017	0.016	0.013	0.015	0.004	0.008	0.012	**0.022**	0.015
Yeast	0.151	**0.171**	0.131	0.112	0.168	0.076	0.051	0.145	0.146	0.077
Human	0.163	**0.174**	0.141	0.167	0.140	0.105	0.122	0.127	0.102	0.076
Birds	0.500	**0.522**	0.516	0.490	0.515	0.457	0.491	0.503	0.327	0.435
Genbase	**0.976**	0.968	0.967	0.965	0.937	0.970	0.967	0.000	0.926	0.000
Medical	0.658	0.671	0.650	0.641	**0.674**	0.654	0.637	0.085	0.222	0.000
NusWide	0.188	**0.213**	0.211	0.165	0.210	0.109	0.108	0.189	0.028	0.000
Stackex coffee	0.085	0.030	0.032	**0.088**	0.014	DNF	0.058	0.000	0.022	0.000
CAL500	**0.000**	**0.000**	**0.000**	**0.000**	DNF	**0.000**	**0.000**	DNF	**0.000**	**0.000**

Table 13
Results of ↑MaP for EAGLET and state-of-the-art EMLCs. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT	D3C	EMLS
Emotions	0.681	0.685	0.704	0.640	0.673	0.588	0.588	0.647	**0.724**	0.591
Reuters1000	**0.253**	0.170	0.134	0.243	0.244	0.165	0.208	0.137	0.220	0.210
Guardian1000	0.259	0.166	0.133	0.250	**0.272**	0.242	0.202	0.120	0.203	0.215
Bbc1000	**0.375**	0.216	0.214	0.353	0.359	0.248	0.267	0.179	0.349	0.238
3s-inter3000	0.120	0.139	0.094	0.144	0.090	**0.165**	0.157	0.117	0.033	0.149
Gnegative	**0.563**	0.495	0.499	0.476	0.473	0.369	0.349	0.406	0.453	0.331
Plant	0.202	0.178	0.189	0.190	0.170	0.145	0.157	0.135	**0.262**	0.163
Water-quality	0.559	0.556	0.573	0.536	0.281	0.500	0.498	0.522	**0.616**	0.553
Yeast	0.512	0.495	0.515	0.463	0.505	0.389	0.392	0.465	**0.547**	0.415
Human	0.241	0.222	0.224	0.206	0.211	0.143	0.171	0.182	**0.244**	0.169
Birds	0.422	0.431	0.420	0.398	0.330	0.318	0.408	**0.432**	0.298	0.355
Genbase	**0.930**	0.923	0.921	0.925	0.768	0.921	0.929	0.217	0.907	0.835
Medical	0.637	0.645	**0.647**	0.645	0.625	0.627	0.646	0.379	0.463	0.427
NusWide	0.171	0.167	0.174	**0.175**	0.153	0.127	0.158	0.154	0.088	0.053
Stackex coffee	**0.641**	0.626	0.627	0.639	0.609	DNF	0.635	0.597	0.578	0.158
CAL500	0.190	0.181	0.187	**0.194**	DNF	0.176	0.177	DNF	0.190	0.175

EMLCs, such as ECC, EBR, RA*k*EL, EPS, HOMER, MLS, RF-PCT, D3C, and EMLS. Tables 11–15 present the results for HL, SA, MaP, MaR, and MaS, respectively.

For HL, EBR, ECC, and D3C methods performed the best in 7 datasets each, while EAGLET performed best in 3 cases. As the number of labels increases, it can be easily seen that the results were nearer to 0 and also the best results were obtained by several methods. This shows the previously mentioned problem of HL, namely, that it is not the best evaluation measure to assess methods in scenarios when the number of labels is large and only few instances are associated with each. On the other hand, for CAL500, Water-quality and Yeast datasets, which have large cardinality values, the HL tends to be clearly higher than for the rest of the datasets.

For SA, again the best results were spread among different methods, although ECC was the best on 7 datasets, followed by EAGLET, RA*k*EL, and D3C, each being the best in 3 cases. Note, that for example EBR, which previously achieved good performance, got very poor results for SA. This could be due to the fact that EBR does not consider the relationships among labels, so correctly predicting all labels for each example is difficult.

Table 14
Results of ↑MaR for EAGLET and state-of-the-art EMLCs. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT	D3C	EMLS
Emotions	0.638	0.585	0.590	0.627	0.621	0.606	0.575	0.667	0.648	**0.739**
Reuters1000	0.141	0.084	0.045	0.158	0.130	0.158	0.162	0.044	0.031	**0.257**
Guardian1000	0.151	0.077	0.041	0.163	0.142	0.226	0.141	0.040	0.036	**0.251**
Bbc1000	0.157	0.086	0.057	0.177	0.163	0.235	0.164	0.039	0.057	**0.289**
3s-inter3000	0.069	0.069	0.031	0.098	0.053	0.165	0.166	0.035	0.004	**0.203**
Gnegative	0.377	0.341	0.299	0.386	0.298	0.374	0.360	0.255	**0.560**	0.522
Plant	0.081	0.069	0.051	0.101	0.046	0.140	0.165	0.042	0.234	**0.345**
Water-quality	0.503	0.519	0.465	0.525	0.148	0.579	0.453	**0.587**	0.492	0.555
Yeast	0.384	0.389	0.351	0.405	0.358	0.406	0.394	0.402	0.402	**0.544**
Human	0.092	0.086	0.066	0.118	0.064	0.141	0.157	0.057	0.242	**0.297**
Birds	0.247	0.222	0.201	0.246	0.198	0.295	0.271	0.217	**0.641**	0.440
Genbase	**0.938**	0.929	0.926	0.931	0.759	0.912	0.935	0.216	0.921	0.885
Medical	0.640	**0.646**	0.641	0.645	0.600	0.598	**0.646**	0.335	0.640	0.516
NusWide	0.139	0.137	0.136	0.147	0.134	0.125	0.152	0.134	0.131	**0.364**
Stackex coffee	**0.636**	0.619	0.620	0.633	0.611	DNF	0.631	0.597	0.598	0.282
CAL500	0.139	0.132	0.124	0.162	DNF	0.224	0.157	DNF	0.177	**0.379**

Table 15
Results of ↑MaS for EAGLET and state-of-the-art EMLCs. Best results for each dataset are marked in bold. DNF values indicate that the method did not finish its execution.

	EAGLET	ECC	EBR	RAkEL	EPS	HOMER	MLS	RF-PCT	D3C	EMLS
Emotions	0.859	0.861	0.881	0.834	0.856	0.805	0.810	0.828	**0.886**	0.756
Reuters1000	0.912	0.924	0.952	0.896	0.910	0.793	0.867	0.964	**0.978**	0.776
Guardian1000	0.917	0.929	0.958	0.897	0.910	0.822	0.863	0.969	**0.986**	0.773
Bbc1000	0.928	0.936	0.964	0.911	0.912	0.817	0.865	0.974	**0.979**	0.779
3s-inter3000	0.891	0.907	0.952	0.863	0.918	0.811	0.794	0.967	**0.987**	0.723
Gnegative	0.963	0.964	**0.973**	0.949	0.968	0.922	0.922	0.964	0.945	0.869
Plant	0.974	0.974	**0.985**	0.959	0.982	0.915	0.917	0.979	0.957	0.830
Water-quality	0.773	0.759	0.800	0.735	**0.940**	0.662	0.741	0.685	0.816	0.750
Yeast	0.788	0.774	0.804	0.761	0.786	0.727	0.743	0.746	**0.806**	0.675
Human	0.974	0.969	**0.981**	0.954	0.975	0.927	0.927	0.972	0.948	0.851
Birds	0.989	0.993	**0.995**	0.986	0.992	0.970	0.985	0.991	0.915	0.956
Genbase	**1.000**	**1.000**	**1.000**	**1.000**	0.999	**1.000**	**1.000**	**1.000**	0.998	0.873
Medical	0.995	0.994	0.995	0.995	0.994	0.995	0.995	**0.999**	0.969	0.726
NusWide	0.995	0.997	**0.998**	0.993	**0.998**	0.980	0.987	0.997	0.965	0.571
Stackex coffee	0.996	0.998	0.998	0.997	0.994	DNF	0.996	**1.000**	0.986	0.397
CAL500	0.913	0.916	**0.924**	0.890	DNF	0.814	0.887	DNF	0.869	0.654

In MaP, both EAGLET and D3C obtained the best results, being the best on 5 datasets each; ECC was not the best method in any case. For MaR, EMLS, which did not obtain a competitive performance in the rest of measures, was the best method in most of the cases. Finally, in MaS, both EBR and D3C obtained the best results on 7 and 6 datasets each, respectively.

In order to determine if the performance of the EMLCs was statistically different, Skillings–Mack's test was performed. Table 16 shows the rankings computed by the Skillings–Mack's test; different algorithms obtain the best ranking value in each measure. As previously introduced, we observe that although EMLS has the best performance for MaR, its performance in the rest of measures is very poor. Also, note that in Tables 11 and 15, D3C seemed to obtain a great performance in HL and MaS measures; however, its ranking for these measures is lower than expected, demonstrating its inconsistent performance, i.e., performing poorly in some of the datasets. In all cases, EAGLET obtains a better ranking value than D3C. Further, EAGLET obtains the best average ranking among all evaluation measures, showing its consistent performance.

The Skillings–Mack's test concluded that there exist significant differences among the performance of methods for all measures, with p-values 2.41E-13 (HL), 1.49E-6 (SA), 1.96E-6 (MaP), 2.18E-7 (MaR), and 4.24E-13 (MaS). Therefore, Holm's test was performed and the results are presented in Table 17. Although EAGLET was the control algorithm in only one case (MaP), for the rest of evaluation measures, EAGLET performed statistically equal to the control algorithm. On the other hand, the rest of EMLCs had significantly worse performance than the control algorithm in at least one evaluation measure. For example, ECC and EBR, which achieved a great performance in some of the evaluation measures, had poorer performance than EMLS in MaR, and D3C performs significantly worse than the control algorithm in three measures. RAkEL, which builds an EMLC similar to the one used in EAGLET, also performed statistically worse than the control algorithm in HL and MaS measures.

Table 16
Average rankings of the Skillings–Mack's test in the comparison among EAGLET and state-of-the-art EMLCs. Last column shows the average ranking among all measures.

	HL	SA	MaP	MaR	MaS	avgRank
EAGLET	3.78	3.34	**2.53**	4.94	4.50	**3.82**
ECC	3.19	**3.03**	4.88	6.28	4.03	4.28
EBR	**2.44**	5.69	4.47	7.59	**2.25**	4.49
RAkEL	5.88	4.72	3.91	3.72	5.88	4.82
EPS	5.09	3.94	5.78	7.56	4.47	5.37
HOMER	8.03	6.50	7.28	4.38	7.94	6.82
MLS	7.59	6.06	6.09	4.41	7.28	6.29
RF-PCT	4.69	7.00	7.47	7.72	3.78	6.13
D3C	4.88	6.72	5.03	5.66	5.19	5.49
EMLS	9.44	8.00	7.56	**2.75**	9.69	7.49

Given the results of the experimental study, we conclude that EAGLET was able to outperform standard MLC methods and also other state-of-the-art EMLCs, which improved the predictive performance of standard methods. Further, EAGLET demonstrated a very consistent performance among all evaluation measures.

Table 17
Results of Holm's post-hoc test comparing EAGLET and state-of-the-art EMLCs. Algorithms marked with "–" are the control algorithm in each measure, while *p*-values in bold typeface indicate that there are significant differences with the control algorithm at 95% confidence.

	HL	SA	MaP	MaR	MaS
EAGLET	≥0.2	≥0.2	–	0.1640	0.1422
ECC	≥0.2	–	0.0857	**0.0058**	0.1922
EBR	–	0.0523	0.1406	**0.0000**	–
RAkEL	**0.0079**	≥0.2	0.1990	≥0.2	**0.0042**
EPS	0.0654	≥0.2	**0.0120**	**0.0000**	0.1422
HOMER	**0.0000**	**0.0072**	**0.0001**	≥0.2	**0.0000**
MLS	**0.0000**	**0.0231**	**0.0052**	≥0.2	**0.0000**
RF-PCT	0.1067	**0.0017**	**0.0000**	**0.0000**	0.1922
D3C	0.0911	**0.0040**	0.0781	**0.0331**	**0.0303**
EMLS	**0.0000**	**0.0000**	**0.0000**	–	**0.0000**

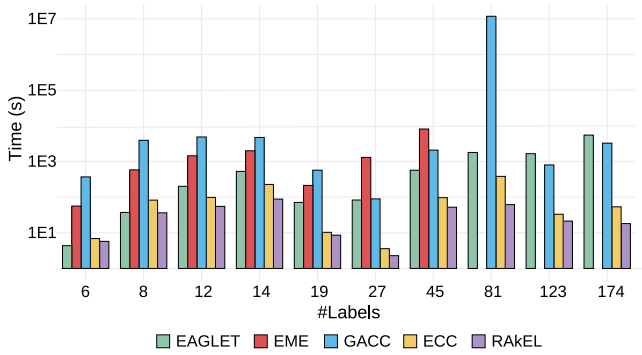


Fig. 6. Comparison of the runtime (in seconds) among MLC methods. The *x*-axis is in logarithmic scale, and results grouped by the number of labels of the datasets. Since EME did not finish its execution in the most complex datasets, its runtime is not shown in these cases.

5.5. Analysis of the runtime

In this section, we analyze the runtime of EAGLET and compare it to other state-of-the art methods. In the supplementary material, the required runtime of all the algorithms is provided. However, in this section we compare the runtime of EAGLET to EME and GACC, which also use evolutionary algorithms, to RA*k*EL, whose operation is similar to the EMLC obtained by EAGLET, and also to ECC, which demonstrated a good performance (being the second EMLC with better average ranking).

In Fig. 6, the runtime of these algorithms is presented. The results are ordered by the number of labels of the data, and for those datasets with the same number of labels (such as Water-quality, Yeast, and Human, all of them with 14 labels), the results were averaged. The *x*-axis is in logarithmic scale, for better presentation of the results. Note that since EME was not able to finish its execution for the most complex datasets, its runtime is not shown in the three last cases.

We first observe as EAGLET has a lower runtime than EME, also considering the reduction in the complexity, which made EAGLET able to obtain an EMLC in reasonable time for complex datasets. Since EME evolves the entire ensemble as individuals, their evaluation is much more complex than just evaluating separate members of the ensemble, as EAGLET does. Therefore, we not only demonstrated that EAGLET achieved a better predictive performance than EME, but also EAGLET is less computationally complex.

EAGLET was able to obtain a model in much less time than GACC, except for the two last cases. However, note that although both use evolutionary algorithms, EAGLET is building an ensemble, and GACC just a single multi-label model, so trying to build an ensemble of GACC members should be much more complex. So, EAGLET not only obtained a significantly better predictive performance than GACC, but also it needs lower runtime in most cases.

Finally, when compared to other EMLCs such as ECC and RA*k*EL, we observe that EAGLET is more complex than them, needing a higher runtime to obtain its model. However, we previously demonstrated than the predictive performance of EAGLET was significantly better than other EMLCs, such as ECC and RA*k*EL, so EAGLET would be a very good option in cases when the runtime of the algorithm is not a main problem.

5.6. Discussion

Although EME had the same objective than EAGLET, the fact of evolving the whole ensemble instead of each of the members of the ensemble, lead to a more difficult optimization of the final ensemble and the members that formed it. EAGLET evolved a population of multi-label classifiers based on projections of the label space, i.e., a population of future hypothetical members of an ensemble; then EAGLET combined a subset of these accurate multi-label classifiers while favoring the construction of a diverse ensemble thanks to the β parameter (see Section 3.6). It has been shown that using a high β value, which leads to selecting more diverse classifiers for the ensemble, EAGLET obtained a better predictive performance (see Section 5.1). Therefore, thanks to the presented method to build the EMLC in EAGLET, it outperformed the predictive performance of EME as well as was able to significantly speed-up its runtime and reduce complexity (see Sections 5.2 and 5.5).

The state-of-the-art methods, including standard MLC methods and state-of-the-art EMLCs, have been assessed on five different evaluation measures and on 16 different datasets. For datasets with a high number of labels and low cardinality, the results in HL for all the methods were similar and near to zero, and there were several ties among algorithms. SA is a widely used measure, but it is so strict that its results should be carefully analyzed. For example, for CAL500, which had as many different labelsets as instances, all methods obtained a value of 0.0 in SA; so only considering this measure will not give much useful information.

On the other hand, using macro-averaged evaluation measures, we give the same importance to all labels. In scenarios with high number of labels and a high imbalance, using other type of evaluation measures could lead to an omission of infrequent labels in their calculation, biasing the results by frequent labels.

In both comparisons against standard MLC methods and state-of-the-art EMLCs, EAGLET obtained a promising and consistent performance. Compared with standard MLC methods, EAGLET was the best method in 2 of the 5 evaluation measures and also obtained the best average ranking value among all measures. EAGLET performed statistically better than the rest of methods in at least 2 of the evaluation measures each. On the other hand, compared with state-of-the-art EMLCs, which have been proven to obtain better results than standard methods, EAGLET also obtained competitive results. In these cases, EAGLET was the control method in only one out of five evaluation measures; however, it again obtained the best average ranking value and also was the only one that obtained statistically same performance than the control algorithm in the rest of evaluation measures at 95% confidence. The rest of state-of-the-art EMLCs performed statistically worse than the control algorithm in at least one evaluation measure.

Therefore, we designed a new method to build ensembles of both accurate and diverse multi-label classifiers, improving previously proposed similar methods and also outperforming state-of-the-art methods in MLC.

6. Conclusions

In this paper we proposed an evolutionary algorithm, EAGLET, focused on creation of an EMLC where each of the members is a multi-label classifier able to predict a subset of k labels. EAGLET considers characteristics of the data such as the imbalance of the dataset when building the ensemble and the relationship among labels in the prediction phase, considering the relationship among small subsets of k labels.

EAGLET evolves a population of multi-label classifiers with subset of labels. Then, it builds an ensemble by combining these multi-label classifiers, considering their predictive performance and favoring selection of diverse members in such a way that all labels appear a similar number of times in the final ensemble.

The experimental study over a set a 16 MLC datasets and 5 evaluation measures has shown the strengths of EAGLET. It outperformed EME, a similar approach of building EMLCs, showing its benefits. EAGLET was significantly better than standard MLC methods. When compared with state-of-the-art EMLCs, although not being the best in several cases, the statistical tests showed that EAGLET's performance was on par with the methods that performed best on each individual measure. EAGLET was the only algorithm that did not perform significantly worse than the control algorithm in each case.

EAGLET has improved the performance of EME, and has shown consistent performance and competitive results when compared with several of state-of-the-art methods.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Jose M. Moyano: Formal analysis, Investigation, Software, Validation, Writing - original draft, Writing - review & editing. **Eva L. Gibaja:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Writing - review & editing. **Krzysztof J. Cios:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Writing - review & editing. **Sebastián Ventura:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Resources, Supervision, Writing - review & editing.

Acknowledgments

This research was supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund, project TIN2017-83445-P. This research was also supported by the Spanish Ministry of Education under FPU Grant FPU15/02948.

References

[1] E. Gibaja, S. Ventura, Multi-label learning: a review of the state of the art and ongoing research, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 4 (6) (2014) 411–444.
[2] A.C. Tan, D. Gilbert, Y. Deville, Multi-class protein fold classification using a new ensemble machine learning approach, Genome Inf. 14 (2003) 206–217.
[3] P. Mangiameli, D. West, R. Rampal, Model selection for medical diagnosis decision support systems, Decis. Support Syst. 36 (3) (2004) 247–259.
[4] H.-J. Lin, Y.-T. Kao, F.-W. Yang, P.S.P. Wang, Content-based image retrieval trained by adaboost for mobile application, Int. J. Pattern Recognit. Artif. Intell. 20 (04) (2006) 525–541.
[5] J. Sun, J. Lang, H. Fujita, H. Li, Imbalanced enterprise credit evaluation with DTE-sbd: Decision tree ensemble based on SMOTE and bagging with differentiated sampling rates, Inform. Sci. 425 (2018) 76–91.
[6] J. Bi, C. Zhang, An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme, Knowl.-Based Syst. 158 (2018) 81–93.
[7] C. Zhang, J. Bi, S. Xu, E. Ramentol, G. Fan, B. Qiao, H. Fujita, Multi-imbalance: An open-source software for multi-class imbalance learning, Knowl.-Based Syst. 174 (2019) 137–143.
[8] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (3) (2011) 335–359.
[9] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multi-label classification, IEEE Trans. Knowl. Data Eng. 23 (7) (2011) 1079–1089.
[10] Y. Wang, S. Ding, X. Xu, W. Jia, The multi-tag semantic correlation used for micro-blog user interest modeling, Eng. Appl. Artif. Intell. 85 (2019) 765–772.
[11] M. Wozniak, M. Graña, E. Corchado, A survey of multiple classifier systems as hybrid systems, Inf. Fusion 16 (2014) 3–17, Special Issue on Information Fusion in Hybrid Intelligent Fusion Systems.
[12] Álvar Arnaiz-González, J.-F. Díez-Pastor, J.J. Rodríguez, C. García-Osorio, Local sets for multi-label instance selection, Appl. Soft Comput. (ISSN: 1568-4946) 68 (2018) 651–666.
[13] O. Reyes, C. Morell, S. Ventura, Scalable extensions of the ReliefF algorithm for weighting and selecting features on the multi-label learning context, Neurocomputing 161 (2015) 168–182.
[14] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Džeroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognit. 45 (9) (2012) 3084–3104.
[15] J.M. Moyano, E.L. Gibaja, K.J. Cios, S. Ventura, Review of ensembles of multi-label classifiers: Models, experimental study and prospects, Inf. Fusion (ISSN: 1566-2535) 44 (2018) 33–45.
[16] G. Tsoumakas, I. Partalas, I. Vlahavas, A taxonomy and short review of ensemble selection, in: Workshop on Supervised and Unsupervised Ensemble Methods and their Applications, 2008, pp. 1–6.
[17] L. Kuncheva, C. Whitaker, C. Shipp, R. Duin, Limits on the majority vote accuracy in classifier fusion, Pattern Anal. Appl. 6 (1) (2003) 22–31.
[18] Y. Bi, The impact of diversity on the accuracy of evidential classifier ensembles, Internat. J. Approx. Reason. 53 (4) (2012) 584–607.
[19] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv. 47 (3) (2015).
[20] S. Ventura, J.M. Luna, Pattern Mining with Evolutionary Algorithms, Springer, 2016.
[21] S.J. Nanda, G. Panda, A survey on nature inspired metaheuristic algorithms for partitional clustering, Swarm Evol. Comput. 16 (2014) 1–18.
[22] H. Faris, M.M. Mafarja, A.A. Heidari, I. Aljarah, A.M. Al-Zoubi, S. Mirjalili, H. Fujita, An efficient binary salp swarm algorithm with crossover scheme for feature selection problems, Knowl.-Based Syst. 154 (2018) 43–67.
[23] M. Taradeh, M. Mafarja, A.A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, H. Fujita, An evolutionary gravitational search-based feature selection, Inform. Sci. 497 (2019) 219–239.
[24] E. Goncalves, A. Plastino, A.A. Freitas, A genetic algorithm for optimizing the label ordering in multi-label classifier chains, in: IEEE 25th International Conference on Tools with Artificial Intelligence, IEEE Computer Society Conference Publishing Services (CPS), 2013, pp. 469–476,
[25] J.M. Moyano, E.L. Gibaja, K.J. Cios, S. Ventura, An evolutionary approach to build ensembles of multi-label classifiers, Inf. Fusion (ISSN: 1566-2535) 50 (2019) 168–180.
[26] J.M. Moyano, E. Gibaja, S. Ventura, An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 2015–2021.
[27] O. Gharroudi, H. Elghazel, A. Aussem, Ensemble multi-label classification: A comparative study on threshold selection and voting methods, in: IEEE International Conference on Tools with Artificial Intelligence, 2015, pp. 377–384.
[28] G. Tsoumakas, I. Katakis, I. Vlahavas, Data Mining and Knowledge Discovery Handbook, Part 6, Springer, 2010, pp. 667–685, Ch. Mining Multi-label Data.
[29] M. Boutell, J. Luo, X. Shen, C. Brown, Learning multi-label scene classification, Pattern Recognit. 37 (2004) 1757–1771.
[30] L. Tenenboim-Chekina, L. Rokach, B. Shapira, Identification of label dependencies for multi-label classification, in: Working Notes of the Second International Workshop on Learning from Multi-Label Data, 2010, pp. 53–60.
[31] J. Read, A pruned problem transformation method for multi-label classification, in: Proceedings of the NZ Computer Science Research Student Conference, 2008, pp. 143–150.
[32] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, in: ECML '09: 20th European Conference on Machine Learning, 2009, pp. 254–269.
[33] M. Zhang, L. Wu, LIFT: Multi-label learning with label-specific features, IEEE Trans. Pattern Anal. Mach. Intell. 37 (1) (2015) 107–120.

[34] H. Blockeel, L.D. Raedt, J. Ramon, Top-down induction of clustering trees, in: Proceedings of the Fifteenth International Conference on Machine Learning, in: ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 55–63.
[35] M.-L. Zhang, Z.-H. Zhou, A k-nearest neighbor based algorithm for multi-label classification, in: Proceedings of the IEEE International Conference on Granular Computing (GrC), Vol. 2, The IEEE Computational Intelligence Society, Beijing, China, 2005, pp. 718–721,
[36] M.-L. Zhang, Z.-H. Zhou, Multi-label neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (2006) 1338–1351.
[37] N. Zhang, S. Ding, J. Zhang, Multi layer ELM-RBF for multi-label learning, Appl. Soft Comput. 43 (2016) 535–545.
[38] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, I. Vlahavas, Correlation-based pruning of stacked binary relevance models for multi-label learning, in: 1st International Workshop on Learning from Multi-Label Data (MLD'09), 2009, pp. 101–116.
[39] G. Tsoumakas, I. Katakis, I. Vlahavas, Effective and efficient multilabel classification in domains with large number of labels, in: Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08), 2008, pp. 53–59.
[40] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Ensembles of multi-objective decision trees, in: European Conference on Machine Learning, 2007, pp. 624–631.
[41] C. Lin, W. Chen, C. Qiu, Y. Wu, S. Krishnan, Q. Zou, LibD3C: Ensemble classifiers with a clustering and dynamic selection strategy, Neurocomputing 123 (2014) 424–435.
[42] B. Liu, G. Tsoumakas, Synthetic oversampling of multi-label data based on local label distribution, 2019, CoRR abs/1905.00609.
[43] K. Dembczyński, W. Waegeman, W. Cheng, E. Hüllermeier, On label dependence and loss minimization in multi-label classification, Mach. Learn. 88 (1) (2012) 5–45.
[44] J. Su, H. Zhang, A fast decision tree learning algorithm, in: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1, in: AAAI'06, 2006, pp. 500–505.
[45] J.M. Moyano, E.L. Gibaja, S. Ventura, MLDA: a tool for analyzing multi-label datasets, Knowl.-Based Syst. 121 (2017) 1–3.
[46] T. Zhou, D. Tao, X. Wu, Compressed labeling on distilled labelsets for multi-label learning, Mach. Learn. 88 (1) (2012) 69–126.
[47] R.B. Pereira, A. Plastino, B. Zadrozny, L.H. Merschmann, Correlation analysis of performance measures for multi-label classification, Inf. Process. Manage. 54 (3) (2018) 359–369.
[48] S. Ventura, C. Romero, A. Zafra, J.A. Delgado, C. Hervás, JCLEC: a Java framework for evolutionary computation, Soft Comput. 12 (4) (2008) 381–392.
[49] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: A Java library for multi-label learning, J. Mach. Learn. Res. 12 (2011) 2411–2414.
[50] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: An update, SIGKDD Explor. Newsl. (ISSN: 1931-0145) 11 (1) (2009) 10–18.
[51] L. Rokach, A. Schclar, E. Itach, Ensemble methods for multi-label classification, Expert Syst. Appl. 41 (16) (2014) 7507–7523.
[52] M. Chatfield, A. Mander, The skillings–mack test (friedman test when there are missing data), Stata J. 9 (2) (2009) 299–305.
[53] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. (1979) 65–70.
[54] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.
[55] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (Dec) (2008) 2677–2694.

Chapter 6

Discussion and conclusions

Throughout the development of this Ph.D. thesis, a number of contributions to MLC have been made, including a comprehensive study of state-of-the-art EMLCs and the proposal of two approaches to build EMLCs. In Section 6.1, the conclusions derived from this Ph.D. thesis are presented. The thorough study of EMLCs as well as the promising results achieved by the proposed models, suggest that this Ph.D. thesis could serve as basis for further investigation by multi-label classification researchers. Then, Section 6.2 presents some lines of future work, including both the development of models to build EMLCs based on those proposed in this dissertation, as well as novel approaches to build EMLCs.

6.1 Concluding remarks

In this dissertation, we have first presented a study and categorization of EMLCs, and then we proposed two evolutionary methods for building ELMCs. The main contributions of this work are already published in three journal papers. The experimental review of EMLCs is carried out in [J4], while the EAs to build EMLCs, EME and EAGLET, are in [J5] and [J6] respectively. In addition, we include three more papers associated with the topic of this dissertation. In [J2], a tool for preprocessing and analyzing multi-label datasets is developed. In [C13] and [C14], two preliminary evolutionary approaches based on CCEAs and G3P are introduced.

Following, we highlight the contributions and conclusions obtained from this work.

6.1.1 Experimental review and categorization of EMLCs

The review of the literature on EMLCs provides a thorough review of the state-of-the-art algorithms. Using it, we proposed a taxonomy where the EMLCs are categorized using two criteria. First, EMLCs are categorized depending on the multi-label classifier they are based on to build the ensembles, such as BR, LP, PCT, or independent of the multi-label classifier. Second, EMLCs are categorized depending on the procedure they use to generate diversity in the ensemble, such as selecting different subsets of instances, subsets input features, subsets output labels, or maybe using different hyperparameters for the learning algorithms. This taxonomy presents, to the best of our knowledge, the first taxonomy to specifically categorize EMLCs.

In addition to the categorization, in the first study we carried out several experiments to determine, depending on the characteristics of a given problem, which EMLC or group of EMLCs perform better. It has been demonstrated that in cases where the imbalance among labels is small, ELP obtains a great performance. However, one of the main drawbacks of ELP is that it usually generates very imbalanced subproblems at each member. Consequently, it is not surprising that in scenarios with moderately and very imbalanced output spaces, RAkEL, which also considers the relationships among subsets of labels but leading to much less imbalanced subproblems, performs the best.

Furthermore, the experimental results have indicated that both RAkEL and ECC perform well on datasets where the dependency among labels is low and medium. Both of them are able to model the relationship among groups of labels: RAkEL by selecting subsets of the labels for each member, and ECC by chaining the binary models and considering the relationship with labels that are previously in the chain. On the other hand, ELP considers all labels at once, being able to exploit better and more exhaustively the relationships among labels, thus performing better on highly dependent datasets.

In terms of efficiency of the EMLCs, there exist some EMLCs that are very fast, such as CBMLC, but which performance is very poor. Other methods such as CDE

are extremely complex and not able to finish its execution in a reasonable time. Furthermore, ECC, which achieves good results on several datasets, is the second most complex method.

Overall, considering all datasets and all evaluation metrics, ECC obtains the best performance, closely followed by RA*k*EL. It is worth mentioning that EPS, which combines good performance and efficiency, can be a good option if fast but also accurate classifier is needed.

6.1.2 Evolving the entire ensemble as individual

In the first evolutionary approach to build EMLCs, EME encodes an entire ensemble in each of the individuals. The population of EMLCs is evolved towards more promising combinations of base classifiers, looking for an optimal structure.

Unlike state-of-the-art EMLCs, EME considers both the relationship among the labels, imbalance, and high dimensionality of the output space when building the ensemble. EME considers the dimensionality of the output space by selecting small subsets of k labels for each member, thus also obtaining base classifiers that deal with less imbalanced and less complex models.

In addition, EME also considers the imbalance when evaluating an EMLC to assess its quality. It is evaluated not only considering its predictive performance, but also the number of times that each label appears in the ensemble, so as to obtain an accurate EMLC which also considers all labels equally, regardless of their frequency. In order to consider the relationships among labels in the evolution, the mutation operator is more likely to obtain individuals with more related subset of labels than with labels that are not very related among them.

The fact of evolving the EMLC, looking for promising combinations of k-labelsets instead of selecting them randomly, resulted in EME outperforming state-of-the-art EMLCs. As shown, although EME has not the best performance in all cases, it is the only method that does not perform significantly worse than the best method in any metric, thus demonstrating its consistency in a wide range of scenarios.

6.1.3 Evolving separate members of the ensemble

In our second approach to build EMLCs, instead of encoding the entire ensemble as in EME, EAGLET encodes separate members as individuals, which are future hypothetical members for the EMLC. The fact of evolving the members of the ensemble independently enables EAGLET to assess the performance of each member separately, thus being easier to combine accurate classifiers in the ensemble.

Together with the evolution of the separate members, the process of selecting those that form the ensemble have obtained competitive results. The EMLC is built by iteratively selecting the members that maximize both their individual performance and the diversity of the ensemble. Besides, it has been demonstrated that giving more weight to the selection of diverse members than accurate ones, leads to an improvement on the predictive performance of the final ensemble.

By following this new evolutionary process, EAGLET not only drastically reduces the computational complexity of EME, but it also outperforms EME and the rest of the state-of-the-art EMLCs. This improvement has been mainly because it is easier to evolve separate classifier towards more accurate ones, as well as by the reduction of the time needed to evaluate each individual.

6.1.4 Open-source code

In order to increase visibility of our contribution to the scientific community, we have released the code of every algorithm and tool developed in the course of this dissertation. The codes are on GitHub under GPLv3 license.

The aim of publicly providing the codes is two-fold: 1) facilitate reproducibility of the results, as well as use it on new datasets; and 2) any researcher interested in modifying or adapting the algorithms can modify it for their future research. We have not only made available the code of EME and EAGLET, but also of two other preliminary evolutionary approaches: CCEA [C13] and G3P-kEMLC [C14]; these two methods are described in Sections 7.2 and 7.3, respectively.

We also made publicly available the code of two different tools. A library to execute the Mulan algorithms from command-line interface, which was used to

carry out different experiments. Having this code, any researcher who aims to execute these methods can download the library and use it, or extend it with new features, for example include other algorithms and evaluation measures. The code of MLDA [J2], a tool for analyzing and preprocessing multi-label datasets was released too. This tool is described in more detail in Section 7.1.

The repositories resulting from this dissertation are listed below.

- **ExecuteMulan.** https://github.com/kdis-lab/ExecuteMulan
- **EME.** https://github.com/kdis-lab/EME
- **EAGLET.** https://github.com/kdis-lab/EAGLET
- **MLDA.** https://github.com/i02momuj/MLDA
- **CCEA.** https://github.com/kdis-lab/CCEA_EMLCs
- **G3P-kEMLC.** https://github.com/kdis-lab/G3P-kEMLC

6.2 Future work

Although two EAs to build EMLCs have been developed within the scope of this Ph.D. thesis, we consider that there is still work to do to improve both the predictive performance and the efficiency of these models, as well as to find new structures for the EMLCs or approaches to build them. Following, we highlight some lines of future work.

Using variable *k* values. In both EME and EAGLET, as well as RA*k*EL does, each member of the ensemble is focused on predicting a subset of k labels. In all of them, k is a parameter of the algorithm, and fixed for all multi-label classifiers. Usually, $k = 3$ is used, as proposed in [53]; however, it would depend on each problem if the labels are better modeled considering a lower or higher number of labels along with it. Therefore, we propose to study the effect that considering a variable value of k in each member would have in the final performance of the ensemble.

A first approximation has been made in [C14], which is then presented in Section 7.3. In this method, different values of k are used in each base classifier, being randomly chosen at the beginning between a range given as parameter. However, these values are fixed at the beginning of the evolution for each base classifier and not modified then. Although it is able to select the most suitable members for the ensemble, it is not able to modify the number of labels considered in each of them.

We consider that the fact of using a variable value of k and automatically adapting it through the evolution would lead to a better predictive performance, since each classifier would select the most appropriate number of labels. Nevertheless, it is not straightforward to apply to previously proposed algorithms, since many critical parts of the EAs must be modified, such as the mutation and crossover operators, as well as some ensemble-specific parameters, as the number of classifiers in the ensemble.

New approaches to select the ensemble members. Since the construction of the EMLCs is one of the key points to obtain a high-performing method, other ways to generate the ensemble should be investigated. In EAGLET, an iterative greedy algorithm to build the ensemble is proposed, which selects in each iteration the classifier that maximizes a combination of accuracy and diversity.

The aim would be to have an EA which is in any way able to both evolve separate members, but also to evolve the structure of the ensemble (instead of using this iterative process to build it). One idea could be to generate an EA in two steps: the first step focused on obtaining a pool of good candidate individuals for the ensemble, and the second step focused on combining these individuals into an ensemble.

Using different populations. So far, we have used EAs involving one population of individuals. However, other types of EAs, such as CCEAs provide the ability of dealing with different subpopulations at the same time. These subpopulations could be each of them focused on a different part of the problem (maybe

considering different subsets of instances or labels), each of them optimizing the classifiers according to its own criteria. Then, to generate the ensemble, individuals from different subpopulations might be used, so the diversity of the EMLC would improve.

A first approach based on a CCEA has been already proposed [C13], and it is presented in Section 7.2. In this method, several subpopulations are used, each of them focused on a different subset of the data. These subpopulations not only evolve separately, but also exchange useful information between them each some generations. However, we think that there still work to do in this way.

Different structures for the EMLC. The EMLCs proposed in this dissertation have both of them the same final structure: the predictions of n multi-label classifiers, each of them considering k labels are combined, where all members have the same weight in the final prediction.

We consider that other types of structures for the EMLC should be investigated. For example, in [C14] we have proposed a preliminary version of a G3P method to build EMLCs, where the EMLC has a tree structure (see Section 7.3). In this way, at each node of the tree, the predictions of children nodes are gathered and combined, while the final ensemble prediction is the one at the root of the tree. Thus, each base classifier of the ensemble does not have the same weight in the final prediction, but it depends on its depth and the number of children of each combination node.

However, in this first approach the k-labelsets for the base multi-label classifiers are randomly generated at the beginning of the evolution and they do not change, but they are combined in an optimal tree-shaped structure. Therefore, we think that further research in this field would be interesting.

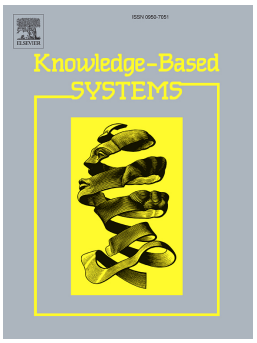
Finally, although up to date we have based all our approaches in the use of k-labelsets for each ensemble member in the same way as RAkEL, other structures as the use of CCs members should be investigated (using all or just part of the labels). The benefit of using CC-based EMLCs is that ECC is usually one

of the best performing EMLCs; however, its optimization would be more difficult since it is not only selecting the labels at each member but also their chaining order.

Chapter 7

Other publications associated with this Ph.D. thesis

7.1 MLDA: A tool for analyzing multi-label datasets



Title	MLDA: A tool for analyzing multi-label datasets
Authors	Jose M. Moyano, Eva L. Gibaja, Sebastián Ventura
Journal	Knowledge-Based Systems
Volume	121
Pages	1 - 3
Year	2017
DOI	10.1016/j.knosys.2017.01.018

IF (JCR 2017)	4.396
Category	Computer Science - Artificial Intelligence
Position	14/132 (Q1)
Cites (27/05/2020)	6 (WoS), 7 (Scopus), 8 (Google Scholar)

Knowledge-Based Systems 121 (2017) 1–3

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Original software publication

MLDA: A tool for analyzing multi-label datasets

Jose M. Moyano[a], Eva L. Gibaja[a], Sebastián Ventura[a,b,*]

[a] Department of Computer Science and Numerical Analysis, University of Cordoba, Spain
[b] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 3 October 2016
Revised 12 January 2017
Accepted 13 January 2017
Available online 14 January 2017

Keywords:
Multi-label learning
Software
Data exploration
Preprocessing
Data analysis
Dataset description

ABSTRACT

The objective of this paper is to present MLDA, a tool for the exploration and analysis of multi-label datasets with both simple and multiple views. MLDA comprises a GUI and a Java API, providing the user with a wide set of charts, metrics, methods for transforming and preprocessing data, as well as comparison of several datasets. The paper introduces the main features of the framework, and introduces its use toward some illustrative examples.

1. Introduction

Multi-label learning (MLL) is a supervised learning paradigm in which a pattern may have associated some classes or labels simultaneously. This learning paradigm has become a challenging research area with an increasing number of papers and application domains, including multimedia classification, medical diagnosis or social network mining, among others [1]. In MLL, datasets usually include characteristics such as imbalance, high dimensionality or relationship among labels [2]. These characteristics could hamper the performance of the algorithms, so characterizing and analyzing the data is essential [3].

To the best of our knowledge, only mldr [4] has been proposed for data characterization in MLL. It is an R package focused on analyzing the imbalance and relationship among labels, which offers both an API and a GUI. Mulan [5] and Meka [6] are two APIs to develop MLL algorithms which also include some basic metrics, data transformations and preprocessing tasks. Finally, *Chekina* et al. [3] provided a wide study on the datasets' characteristics to select the best MLL algorithm, giving rise to a large number of metrics on this sense. For that, a wide number of metrics have been proposed.

With the aim of extending the offer of data characterization tools we have develop the MLDA tool, a GUI and an API for characterization, exploration and analysis of multi-label and multi-view multi-label (MVML) datasets. MLDA includes all the functionality of previous works, also expanding the metrics to analyze the imbalance and dependences among labels as well as different preprocessing methods. Moreover, MLDA includes new tools to convert dataset's format, to analyze MVML data and to compare several datasets.

The rest of the manuscript is organized as follows: Section 2 presents the software framework, Section 3 shows some illustrative examples and finally Section 4 presents some conclusions.

2. Software framework

The MLDA tool code and its documentation (javadoc and user guide) are available at the GitHub repository https://github.com/i02momuj/MLDA. The software has been released under the GPLv3 license.

2.1. Software architecture

Both MLDA GUI and API have been implemented in Java and have been built over Mulan and Weka [7]. The libraries JGraphX [8] and JFreeChart [9] have been used to represent the charts. The API class diagram is shown in Fig. 1. The *base* package includes the base classes from which all the metrics inherit; *metricsTaxonomy* package enables to represent the taxonomy of metrics; *utils* package includes extra methods to calculate metrics; and *dimensional-*

* Corresponding author.
E-mail address: sventura@uco.es (S. Ventura).

http://dx.doi.org/10.1016/j.knosys.2017.01.018
0950-7051/

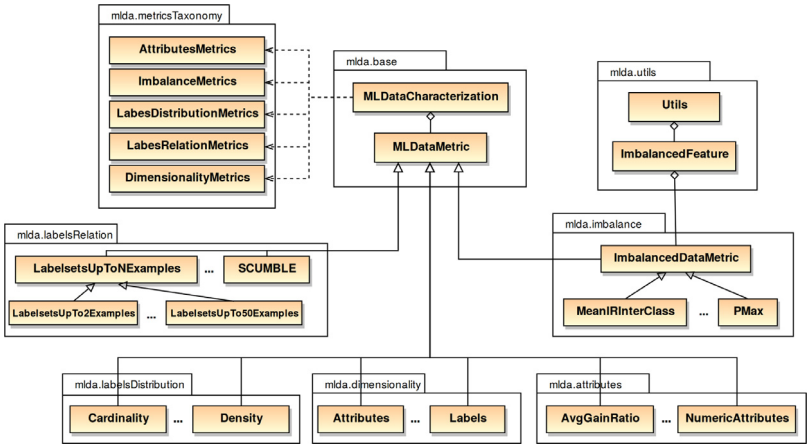
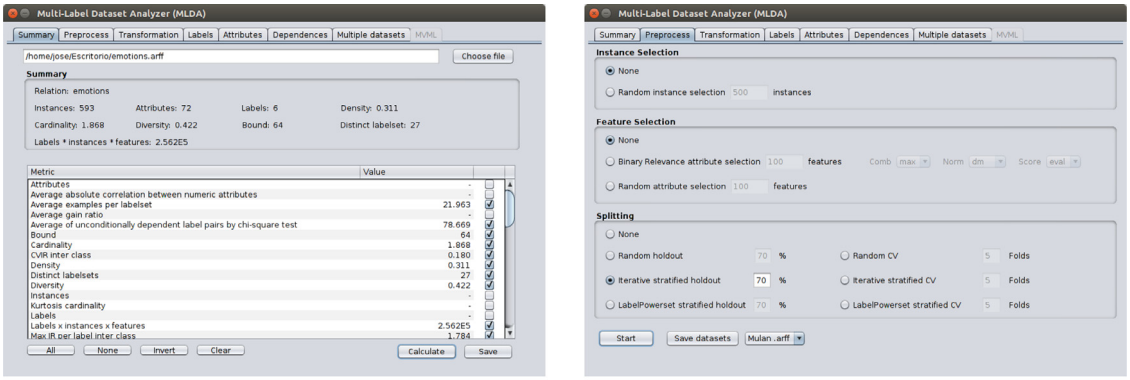
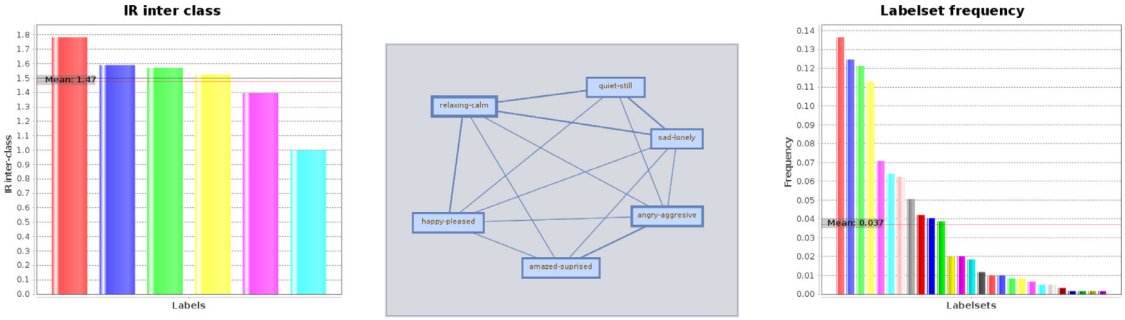


Fig. 1. MLDA class diagram.



(a) *Summary tab* (b) *Preprocess tab*

Fig. 2. Main characteristics and preprocessing.



(a) *IR inter-class per label* (b) *Co-occurrence graph* (c) *Frequency of labelsets*

Fig. 3. Measuring imbalance and relationship among labels.

ity, labelsRelation, labelsDistribution, imbalance and *attributes* packages include the implementation of the corresponding metrics.

2.2. Software functionalities

MLDA provides a wide set of 57 metrics, which can be saved in different file formats (*csv, txt, tex* and *arff*) for further analysis. It includes specific metrics and charts to measure the imbalance of the labels, such as histograms and boxplot diagrams with the labels distribution, and Imbalance Ratio for labels (IR both inter-class and intra-class) and combinations of labels (also called labelsets). It also includes Chi and Phi coefficients, as well as co-occurrence and heatmap graphs to analyze the relationships among labels. Furthermore, MLDA enables to compare several datasets.

Additionally, MLDA includes different instance and feature selection methods. It can also perform data partition in both hold-out and *k*-folds, including random and stratified partitioning specific for MLL. MLDA is able to load, transform and save datasets in both Mulan and Meka file formats. It includes some transformation methods which generate one or more single-label datasets. For MVML datasets, MLDA also shows a set of MVML specific metrics and allows to save a set of user-selected views. Finally, the API includes the full set of metrics included in the GUI and a framework to implement new metrics.

Table 1
Software metadata.

Nr.	(executable) Software metadata description	Please fill in this column
S1	Current software version	1.2.2
S2	Permanent link to executables of this version	https://github.com/i02momuj/MLDA/releases/tag/1.2.2
S3	Legal Software License	GPLv3
S4	Computing platform/Operating System	Linux, Microsoft Windows, OS X
S5	Installation requirements & dependencies	Java version 1.8 or higher
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	https://github.com/i02momuj/MLDA/blob/master/doc/MLDA_Doc.pdf
S7	Support email for questions	sventura@uco.es

Table 2
Code metadata.

Nr.	Code metadata description	Please fill in this column
C1	Current code version	v1.117
C2	Permanent link to code/repository used of this code version	https://github.com/i02momuj/MLDA/releases/tag/1.2.2
C3	Legal Code License	GPLv3
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Java
C6	Compilation requirements, operating environments & dependencies	Java version 1.8 or higher
C7	If available Link to developer documentation/manual	https://github.com/i02momuj/MLDA/releases/download/1.2.2/javadocs.zip
C8	Support email for questions	sventura@uco.es

3. Illustrative examples

In this section, an example of use of MLDA with emotions dataset is shown (available at https://www.uco.es/grupos/kdis/kdiswiki/index.php/Resources). Its main characteristics can be obtained loading the dataset with MLDA (Fig. 2(a)). It provides researcher with a first characterization of the dataset. Usually train and test or *k*-folds partitions are needed to run the algorithms, which can be generated by the *Preprocess* tab (2(b)).

As many algorithms are influenced by the imbalance of labels, it might be analyzed in order to use a rebalancing technique or not. The *Labels* tab shows the *IR inter-class* chart (Fig. 3(a)). As some labels has an $IR > 1.5$, a rebalancing technique could be necessary. In order to use an algorithm either considering label relationships or not, the *Dependences* tab provides a co-occurrence graph, which shows the co-occurrence between each pair of labels (Fig. 3(b)). As all labels are related, using a technique that takes into account the relationship among labels might be more recommendable. Finally, the frequency of the labelsets (Fig. 3(c)) also could increase the complexity of certain algorithms [1]. As many labelsets are very infrequent, an algorithm that prune infrequent labelsets could be recommendable.

4. Conclusions

In this paper MLDA have been proposed, including a GUI tool and a Java API, which is a useful tool in the MLL research field since it enables to characterize, analyze and preprocess MLL and MVML datasets in an easy way. The analysis of the datasets are highly important since it helps researchers to select the most appropriate algorithm depending on the characteristics of the dataset.

Required Metadata

Current executable software version

Table 1

Current code version

Table 2

Acknowledgements

This research was supported by the Spanish Ministry of Education under FPU Grant FPU15/02948, by the Spanish Ministry of Economy and Competitiveness, Project TIN-2014-55252-P, and by FEDER funds.

References

[1] E. Gibaja, S. Ventura, Multi-label learning: a review of the state of the art and ongoing research, WIREs Data Mining Knowl Discov 2014. doi:10.1002/widm.1139.
[2] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv. 47 (3) (2015).
[3] L. Chekina, L. Rokach, B. Shapira, Meta-learning for selecting a multi-label classification algorithm, 2011, pp. 220–227.
[4] F. Charte, D. Charte, Working with multilabel datasets in r: the mldr package, R. J. 7 (2) (2015) 149–162.
[5] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: a java library for multi-label learning, J. Mach. Learn. Res. 12 (2011) 2411–2414.
[6] Meka: A multi-label extension to weka, (http://meka.sourceforge.net/) Last access: 16-11-2016.
[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[8] Jgraphx, (https://www.jgraph.com), Last access: 16-11-2016.
[9] Jfreechart, (http://www.jfree.org/jfreechart/), Last access: 16-11-2016.

7.2 Generating ensembles of multi-label classifiers using cooperative coevolutionary algorithms

Title	Generating ensembles of multi-label classifiers using cooperative coevolutionary algorithms
Authors	Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastián Ventura
Conference	European Conference on Artificial Intelligence (ECAI 2020)
Status	Accepted paper
Year	2020

Generating Ensembles of Multi-Label Classifiers Using Cooperative Coevolutionary Algorithms

Jose M. Moyano[1] and **Eva L. Gibaja**[2] and **Krzysztof J. Cios**[3] and **Sebastián Ventura**[4]

Abstract. Multi-label classification deals with problems where each of the data instances has several labels associated with it. Although many ensemble-based approaches for multi-label classification have been proposed, several of them do not take into account intrinsic characteristics of the data during their design. In this paper we present a cooperative coevolutionary algorithm which considers such specific characteristics to build an ensemble of accurate and diverse multi-label classifiers. The algorithm evolves several subpopulations simultaneously, each using a different subset of the training data. Also, each individual is focused only on a small subset of labels. These two characteristics provide greater diversity of members to generate the ensemble. As it evolves separate members, we also define a procedure to build an ensemble given the individuals. The experimental study comparing the proposed method to the state-of-the-art in multi-label classification using thirteen datasets and five evaluation metrics demonstrated that the developed cooperative coevolutionary algorithm performed consistently and statistically better than the other methods.

1 INTRODUCTION

Multi-Label Classification (MLC) is a classification paradigm capable of dealing with problems where each of the instances of the data may have several labels associated with it simultaneously, unlike traditional classification, where each example has only one class associated with it. For example, in medical diagnosis, a patient can have a few diseases at the same time [21]. The MLC paradigm has been successfully applied not only to medically related problems, but also to multimedia annotation [23], legal documents categorization [11], and prediction of sub-cellular locations of proteins [27]. The fact of dealing with several labels simultaneously, leads to new challenges that need to be tackled, such as modeling the dependencies among labels, and dealing with the data imbalance and high-dimensionality of the output space.

Existing MLC methods are focused on dealing with some or all of these challenges [20, 25]. We focus on the Ensembles of Multi-Label Classifiers (EMLCs), which combine the predictions of many multi-label classifiers, which leads to better performance [20, 18, 9]. Although ensemble models outperform single classifiers, the classifiers combined into the ensemble should not only be accurate but also diverse [26, 1]. Further, although EMLCs are usually able to deal with the different characteristics of the multi-labeled data, such as the relationship among labels, imbalance, and high dimensionality of the output space, many of them do not consider all of them when building the ensemble [13]. For example, RAndom *k*-labELset (RA*k*EL) [25] is able to deal with the relationship among labels, but it just selects random subsets of labels, without considering any of the characteristics of the data for selecting them (more about it in Section 2.2).

One of the ways that have been successfully used for building ensemble learners is the use of Evolutionary Algorithms (EAs) [14, 13]. EAs are biology-inspired search algorithms [4], and they provide an optimal framework for solving the problem of the member selection for the ensemble. Specifically, the Evolutionary Multi-label Ensemble (EME) method [13] proposes an evolutionary algorithm to build EMLCs where each of the individuals of the population is an EMLC. EME not only deals with the three main characteristics of multi-label data, but takes them into account when building the ensemble. Further, the fact of evolving the ensembles toward a fitness function based on both the performance and the diversity of the ensemble results in its outperforming to other state-of-the-art MLC methods.

Although classic EAs have shown good performance in solving optimization problems, several extensions of EAs have been proposed to improve their performance; example are Cooperative Co-Evolutionary Algorithms (CCEAs) [16]. The main difference between EAs and CCEAs is that while in EAs there is just one population of individuals, in CCEAs there are several subpopulations at the same time. Also, individuals in an EA usually represent a full solution to the problem, while in CCEAs, the individuals of each subpopulation usually represent only a partial solution to the problem; the final solution is obtained by combination of individuals from several subpopulations. Further, in CCEAs, individuals not only compete among them (as in traditional EAs), but also cooperate among them, for example either obtaining a full solution as combination of some of the individuals, or sharing useful information among subpopulations.

CCEAs were first proposed because of the need for representing and evolving complex structures. Taking into account the complexity and difficulty of selecting the most appropriate members for the ensemble, the aim of this paper is to propose a CCEA for the generation of EMLCs. The method focuses on building an EMLC where each individual is a different member of the ensemble (unlike in EME, where each individual is the entire ensemble), and where each member is focused only on a small subset of the labels. Further, individuals of each subpopulation use a different subset of the data. The fact of focusing each individual only on a subset of labels allows the model to take into account the relationship among labels but in a less complex way. This, plus the use of different subpopulations over

[1] University of Córdoba, Spain, email: jmoyano@uco.es
[2] University of Córdoba, Spain, email: egibaja@uco.es
[3] Virginia Commonwealth University, U.S.A., and Polish Academy of Sciences, Gliwice, Poland, email: kcios@vcu.edu
[4] University of Córdoba, Spain, email: sventura@uco.es

different data subsets allows for greater diversity for the ensemble. As each individual is a different member of the ensemble, we also propose a method for communicating between subpopulations and building the final solution, the EMLC.

The experimental study carried out over thirteen multi-label datasets and using five evaluation metrics demonstrated that the proposed CCEA performed more consistently and statistically better than the state-of-the-art EMLCs.

The rest of the article is organized as follows: Section 2 provides background and describes the related MLC work; Section 3 presents the CCEA for building EMLCs; Section 4 describes the experimental studies carried out; Section 5 presents and discusses the results; and Section 6 ends with conclusions.

2 RELATED WORK

In this section, we first formally define MLC, and then present state-of-the-art EMLCs.

2.1 Formal definition of MLC

Let $\mathcal{X} = X_1 \times \cdots \times X_d$ be the d-dimensional input space, and $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ the output space composed by $q > 1$ labels. Let $\mathcal{D}$ be a multi-label dataset composed of m instances, as $\mathcal{D} = \{(\mathbf{x}_i, Y_i) | 1 \leq i \leq m\}$, where each multi-label instance is composed by an input feature vector $\mathbf{x} \in \mathcal{X}$ and a set of relevant labels associated with it $Y \subseteq \mathcal{Y}$. The goal of MLC is to construct a predictive model able to provide a set of relevant labels for an unknown instance. Thus, for each $\mathbf{x}$, a bipartition $\left(\hat{Y}, \overline{\hat{Y}}\right)$ of the label space $\mathcal{Y}$ is provided, where $\hat{Y}$ is the set of relevant labels and $\overline{\hat{Y}}$ the set of irrelevant ones.

Further, an EMLC is defined as a set of n multi-label classifiers, each of them providing prediction $\hat{\mathbf{b}}_j = \{\hat{b_{j1}}, \hat{b_{j2}}, \ldots, \hat{b_{jq}}\}$ for all (or part of) the labels. If each model predicts bipartitions, each b_j is 1 if the label is predicted as relevant and 0 otherwise; however, each of them could also provide confidences, being each b_j a value in $[0, 1]$ range indicating the likelihood of each label to be relevant or not. Then, these predictions are combined in some way; majority voting is the most used but there are several other combining methods [6].

2.2 Ensembles of Multi-Label Classifiers

MLC algorithms are categorized into three groups: problem transformation, algorithm adaptation, and EMLCs [7]. Problem transformation methods transform the multi-label problem into one or several single-label problems, which are then solved using traditional classification methods. Algorithm adaptation methods adapt traditional classification methods to directly handle multi-label data, without the need of transforming the dataset. Finally, EMLCs are methods that combine the predictions of several multi-label classifiers. Given the better performance of ensemble methods over simpler ones, we focus attention on the EMLCs. A thorough description of EMLCs can be found on [12].

Ensemble of Binary Relevances (EBR) [20] is based on Binary Relevance (BR) method [24]. BR builds q independent binary models, one for each of the labels, and thus is not able to model dependencies among them. EBR still is not able to model these dependencies, but tries to improve the performance of BR by combining n BR models, each of them built over a different subset of training data.

Ensemble of Classifier Chains (ECC) combines the predictions of several Classifier Chains (CC) [20]. Each of the CC builds q binary models but in this case they are linked in such a way that the predictions of previous labels in the chain are introduced as additional input features, being able to model some of the dependencies among the labels. ECC, on the other hand, consist of n CCs each built over a different subset of the training dataset and with a different random chain. Although able to model some of the dependencies among labels, ECC does not consider these relationships in building the ensemble, e.g., to select the chains.

Ensemble of Pruned Sets (EPS) is built on top of the Pruned Sets (PS) method [19]. PS is an extension of Label Powerset (LP) [22], which transforms the multi-label problem into a multi-class one, where each of the combinations of labels is considered as a different class. PS works as LP but it prunes the classes whose frequency is below a given threshold, reducing imbalance. EPS is built by combining n PS models, each of them built over different subsets of the training data. Therefore, EPS considers both the imbalance and dimensionality of the output space while building the models; it prunes the infrequent labelsets to obtain less complex models.

RAndom k-labELsets (RAkEL) [25] builds an ensemble of LP methods, where each of them is only focused in a small random subset of k labels (a.k.a. k-labelset). The fact of partitioning the label space into smaller subspaces makes RAkEL able to model the relationships among labels, but deals with less imbalanced and complex problems than when all labels are considered at a time. Although able to deal with these relationships, it does not take them into account when building the ensemble, for example, to select subsets of more related labels.

Finally, Evolutionary Multi-label Ensemble (EME) [13] is an evolutionary algorithm to automatically design EMLCs. In EME, each of the individuals of the population is a complete EMLC, where the operation of the EMLC is similar to RAkEL. However, unlike RAkEL, EME does not just select the k-labelsets randomly, but it evolves towards more promising combinations of labels in each member as well as better combinations of members into the ensemble, leading to an improvement of performance of RAkEL and other state-of-the-art methods. The fact of evolving the entire ensemble as an individual not only made EME computationally more complex but also more difficult to converge to a better solution than if members were evolved independently.

3 COOPERATIVE COEVOLUTIONARY ALGORITHM

In this section, we describe the proposed CCEA for building EMLCs. First, we briefly describe the structure and operation of the EMLC generated. Then, we present the CCEA, describing its main steps, representation and initialization of individuals and subpopulations, the way subpopulations communicate, genetic operators, fitness function, and finally, the method used to generate the EMLC.

3.1 Structure of the EMLC

The EMLC obtained in the CCEA consists of n members, each of them considering only a small subsets of k labels. In this way, each member of the ensemble is able to model the compound dependencies among its k-labelset, leading to less complex and less imbalanced models than when the full set of labels is used. Although any multi-label classifier can be used at each member, we use LP as in [25] and [13].

For an unseen instance, each member gives a bipartition for each of the labels in its *k*-labelset. Figure 1 shows an example of the prediction phase of the EMLC for a given instance. Suppose for example that the first classifier is focused on learning labels λ_2, λ_3, and λ_6, so it gives prediction for only these labels. Then, predictions of all classifiers are gathered and the ratio of positive predictions for each label is calculated; if it is greater than a threshold t, the final prediction of the EMLC is positive, and negative otherwise.

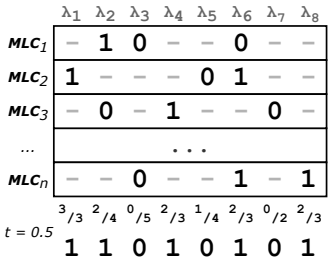


Figure 1: Example of the prediction phase of the EMLC.

3.2 Individuals and initialization

Each individual represents not only the subset of labels that it considers, but also the subpopulation to which it belongs. Therefore, each individual is represented as a binary array, where genes to 1 indicate that the label belongs to its *k*-labelset and genes to 0 that it does not belong; and also with an integer value indicating the index of the subpopulation to which this individual belongs. In Figure 2 we show some examples of individuals. We can see that there are two individuals belonging to each subpopulation $s_i, i \in \{1, 2, 3\}$. For example, individual $I_{1,1}$ will be focused on predicting labels λ_2, λ_3, and λ_6, and built over the subset of the data corresponding to subpopulation s_1. Note that individuals $I_{1,1}$ and $I_{3,1}$, although they focus on predicting the same labels, they are different since they are built over different subsets of the data.

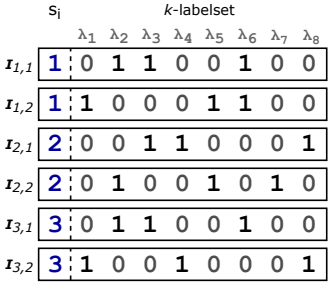


Figure 2: Example of individuals of the CCEA. Each individual includes both the index of the subpopulation as well as the labels present in its *k*-labelset.

At the beginning of evolution, a subset of the data is generated for each subpopulation, in such a way that all individuals of the same subpopulation use always the same data. The individuals are initialized independently for each subpopulation.

Although all labels are required to appear a minimum number of times in each subpopulation, ensuring the use of minority labels, we use their frequency as a proxy of their importance in this process. The expected number of appearances of each label in each subpopulation is calculated using Equation 1, where f_l is the frequency of a given label λ_l, and r is the number of remaining appearances after sharing the minimum number of appearances a_{min} for each label, calculated as $r = k \times subpopSize - q \times a_{min}$. Note that $k \times subpopSize$ is the total number of active bits in the subpopulation. The number of times that a label could appear in the initial subpopulation is upper bounded by the size of the subpopulation.

$$a_l = \max\left(subpopSize, a_{min} + \left\| \frac{f_l}{\sum_{j=1}^{q} f_j} \times r \right\|\right) \quad (1)$$

Individuals are created by activating k randomly selected bits, where labels with higher value of a_l have higher chance to be activated, thus making sure that more frequent labels appear more times in the initial subpopulations. Note that these frequencies are calculated for each subpopulation.

3.3 Steps of the CCEA

Figure 3 shows the main steps of the CCEA. Boxes with double lines indicate that the process is performed independently for each subpopulation. At the beginning, n_s samples of the original training data are selected, where n_s is the number of subpopulations in the algorithm. Then, each subpopulation s_i is initialized (see Section 3.2), and the individuals are evaluated (see Section 3.6). While the maximum number of generations n_g is not reached, individuals are selected by tournament selection, crossover and mutation operators are applied with p_c and p_m probabilities respectively (see Section 3.5), new individuals are evaluated, and the subpopulations for next generation are selected following the same process to generate the ensemble (see Section 3.7). Then, subpopulations communicate between them each n_{gc} generations, obtaining an ensemble (and storing the best so far), and exchanging information between subpopulations (see Section 3.4). The whole process of communication among subpopulations is represented with a dashed box in the figure. Finally, when the maximum number of generations is reached, the best EMLC is returned as the best solution.

3.4 Communication between subpopulations

The communication between subpopulations have two objectives: I) generate a complete solution to the problem, i.e., an EMLC, given the individuals of all subpopulations, and II) transfer good genetic material of individuals from one subpopulation to another at some iterations of the CCEA. This communication is not performed at each iteration, but after n_{gc} iterations; this allows each subpopulation to evolve their own individuals before putting them together with the rest of subpopulations.

In order to generate an EMLC, individuals of all subpopulations are joined, and then the process described in Section 3.7 is followed. After the EMLC is built, it is evaluated using the entire training dataset; it is stored if it is the best ensemble generated so far.

Communication between subpopulations occurs also to exchange information between subpopulations. For that, after n_{gc} generations, individuals of each subpopulation s_i are applied specific crossover and mutation operators, with p_{cc} and p_{mc} probabilities respectively (see Section 3.5). If one individual from s_i is selected for

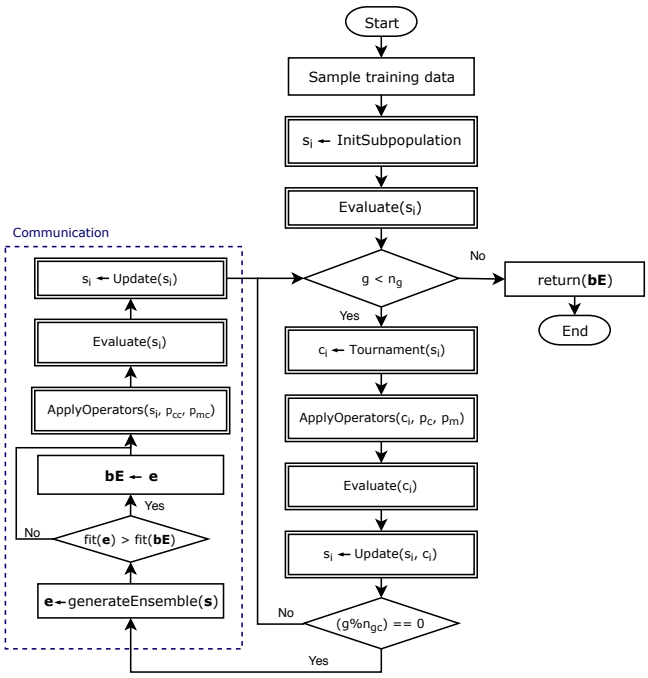


Figure 3: Main steps of the CCEA. Boxes with double lines indicate that the process is performed independently for each subpopulation. The region within the dashed line indicates the communication between subpopulations.

crossover operator, a random individual from a different subpopulation $s_j, i \neq j$ is also selected, thus exchanging genetic material between individuals of different subpopulations. If mutation operator is used, it changes the index of subpopulation without modifying the rest of the genes of the individual; this enables to train an individual with the same k-labelset but in another subpopulation.

3.5 Genetic operators

In this section, we define the crossover and mutation operators used. For that, we need to look at two possible scenarios: the first scenario is when genetic operators are applied to individuals of a given subpopulation, so the index of subpopulation of the individual does not change; the second is when operators are applied to communicate subpopulations, so the index of the individual is considered and it could be modified.

3.5.1 Crossover operator

Given two individuals I_1 and I_2, the crossover operator swaps information relative to their k-labelsets. The child individuals will inherit the subpopulation index of their parents so, independently of the scenario, its operation is the same. In Figure 4, an example of the crossover operator is shown when individuals belongs to different subpopulations. It would be exactly the same if they both belonged to the same subpopulation.

First, the crossover operator creates two sets ds_1 and ds_2 with the positions of genes that are activated in one individual but not in the other (Figure 4a). These sets are shuffled and divided by the midpoint (Figure 4b). Then, two new sets $ds_1{}'$ and $ds_2{}'$ are created with one half of each previous sets (Figure 4c). Finally, crossed individuals I'_1 and I'_2 are created by copying the genes that were identical in both parents and activating the genes of their corresponding sets (Figure 4d). New individuals are always feasible and contain genetic material of both parents.

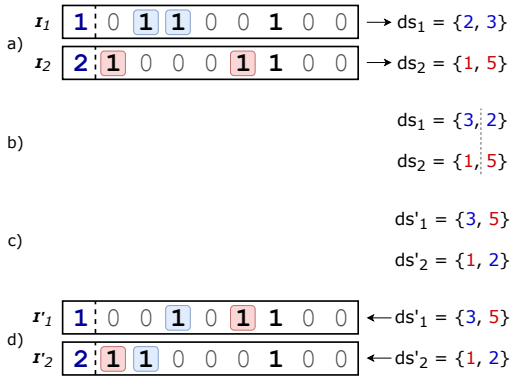


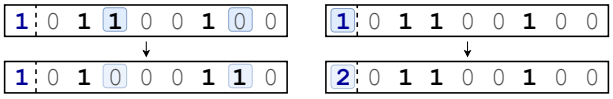
Figure 4: Example of crossover operator.

3.5.2 Mutation operators

We define different mutation operators for each scenario. In both cases, feasible individuals are always obtained after mutation.

The so-called label mutator is used when the mutation operator is applied for a specific subpopulation (Figure 5a). It aims to modify the k-labelset of an individual, randomly selecting one active and one inactive gene, and swapping their values. Unlike the crossover operator, which tries to find new subsets of labels by combining information of existing individuals, mutation operator modifies the k-labelset of a given individual with randomly created genetic material, thus looking for new combinations of labels.

If the mutation operator is applied to communicating subpopulations, we use the subpopulation mutator. In this case, the k-labelset is not modified, but the index of the subpopulation is (Figure 5b). This mutation operator selects a random different subpopulation for the individual, allowing to learn the same combination of labels from the point of view of other subpopulation.



(a) Label mutator (b) Subpopulation mutator

Figure 5: Mutation operators.

3.6 Fitness function

In order to evaluate the fitness of individuals, the corresponding multi-label classifier is built and the Example-based FMeasure (ExF), which is presented in Equation 2, is calculated [8]. FMeasure is a robust evaluation metric used to evaluate classification models in imbalanced scenarios [10]. Although there are several approaches to calculate FMeasure in MLC, ExF evaluates the prediction of each instance as a whole, therefore being able to capture the relationship among labels in its calculation. As our approach is focused on modeling label dependencies of small subsets of labels, we are using ExF as the fitness function.

$$\uparrow \text{ExF} = \frac{1}{m}\sum_{i=1}^{m}\frac{2|\hat{Y}_i \cap Y_i|}{|\hat{Y}_i| + |Y_i|} \quad (2)$$

Each individual is built using the corresponding training data of its subpopulation. The ExF is calculated over the full training dataset, which has two objectives: I) individuals are evaluated over a dataset including unknown instances, allowing to check its generalization ability, and II) all individuals are evaluated over the same data independently of their training datasets, therefore giving a better approximation of how each individual will perform when used in the ensemble. Further, each time an ensemble is built in the communication phase, it is also evaluated by obtaining the ExF of the EMLC over the full training dataset.

As some individuals could appear again in subsequent generations, each classifier is stored in a table along with its fitness. Therefore, if this individual needs to be evaluated again, its fitness is just taken from the table.

3.7 Ensemble generation

The process of generating the ensemble is shown in Algorithm 1. The array with the number of expected votes **eV** is calculated before selecting any member for the ensemble (line 1). This array contains the number of times that each label should be added to the current ensemble; at the beginning this array is calculated spreading votes evenly among all labels. The best individual according to its fitness is selected to initialize the ensemble **e**, and it is removed from p (lines 2-5); then the **eV** array is updated by subtracting one to each label of this individual (line 6). Then, until the ensemble reaches the desired size, the individual that best fits the ensemble, considering both performance and diversity with the current ensemble is selected (lines 7-16). For that, the distance from each individual to the current ensemble is calculated as a weighted distance. This distance is defined in Equation 3, where $[\![\pi]\!]$ returns 1 if predicate π is true and 0 otherwise, and e_i is each of the members of the current ensemble. Also, the weights **w** to calculate the distance are calculated by normalizing the **eV** array in such a way that $\sum_{l=1}^{q} w_l = 1$. This distance gives more weight to labels that are less frequent in the ensemble, favoring the selection of individuals containing them. Then, the individual that maximizes a linear combination between its fitness and the distance is added to the ensemble. The β value could be modified in order to give more importance to the performance of the individuals or to the diversity of the ensemble, thus allowing to generate an ensemble composed of accurate individuals which are diverse. Finally, the ensemble **e** is returned (line 17).

$$d_{ind} = \frac{1}{n'}\sum_{i=1}^{n'}\sum_{l=1}^{q}\left(w_l \times [\![ind^l \neq e_i^l]\!]\right) \quad (3)$$

Algorithm 1 Ensemble generation.

Input: p: set of individuals.
Output: **e**: ensemble of n multi-label classifiers.

1: **eV** $\leftarrow$ calculate expected votes array
2: $b \leftarrow \arg\max_{ind}(fitness_{ind})$
3: **e** $\leftarrow \{b\}$
4: $n' \leftarrow 1$
5: $p \leftarrow p \setminus \{b\}$
6: **eV** $\leftarrow update(\mathbf{eV}, b)$
7: **while** $n' < n$ **do**
8: **for** each individual ind in p **do**
9: $d_{ind} \leftarrow distance(ind, \mathbf{e}, \mathbf{eV})$
10: **end for**
11: $b \leftarrow \arg\max_{ind}(\beta * d_{ind} + (1-\beta) * fitness_{ind})$
12: **e** $\leftarrow$ **e** $\cup \{b\}$
13: $n' \leftarrow n' + 1$
14: $p \leftarrow p \setminus \{b\}$
15: **eV** $\leftarrow update(\mathbf{eV}, b)$
16: **end while**
17: **return e**

4 EXPERIMENTAL STUDY

In this section we describe experimental studies performed, including description of datasets and evaluation metrics used, as well as the experimental settings.

4.1 Datasets

A set of 13 multi-label datasets from different domains was selected to perform our experimental studies[5]. These datasets are shown in Table 1 along with their main characteristics such as the cardinality, i.e., average number of labels associated with each instance (*card*), the average imbalance ratio (*avgIR*), and the ratio of dependent label pairs (*rDep*) [15]. Note that as the number of labels increases, the number of possible different *k*-labelsets also increases, and even more the number of different combinations of *k*-labelsets into an ensemble. We selected datasets ranging from 6 to 123 labels, covering a wide range of complexity.

Table 1: Datasets and their characteristics, including number of instances (m), number of attributes (d), number of labels (q), cardinality (*card*), average imbalance ratio (*avgIR*), and ratio of dependent label pairs (*rDep*). The datasets are ordered by the number of labels.

Dataset	m	d	q	*card*	*avgIR*	*rDep*
Reuters1000	294	1000	6	1.126	1.789	0.667
Guardian1000	302	1000	6	1.126	1.773	0.667
Bbc1000	352	1000	6	1.125	1.718	0.733
GnegativePseAAC	1392	1717	8	1.046	18.448	0.536
PlantPseAAC	978	440	12	1.079	6.690	0.318
Water-quality	1060	16	14	5.073	1.767	0.473
Yeast	2417	103	14	4.237	7.197	0.670
HumanPseAAC	3106	440	14	1.185	15.289	0.418
Birds	645	260	19	1.014	5.407	0.123
Genbase	662	1186	27	1.252	37.315	0.157
Medical	978	1449	45	1.245	89.501	0.039
NusWide[6]	2696	128	81	1.863	89.130	0.087
Stackex coffee	225	1763	123	1.987	27.241	0.017

4.2 Evaluation metrics

For the evaluation of the MLC methods, several evaluation metrics have been used [8]. Hamming loss (HL) evaluates the average number of times a label is incorrectly predicted. It is a minimized metric, and it is defined in Equation 4, where Δ is the symmetric difference between two binary sets. Subset Accuracy (SA), defined in Equation

[5] Datasets were downloaded from the repository in http://www.uco.es/kdis/mllresources

[6] A random selection of the original instances of NusWide cVLAD+ dataset was performed in order to be able to execute it in a reasonable time.

5, is a strict metric that evaluates the ratio of instances whose labelset was perfectly predicted (including all relevant and irrelevant labels).

$$\downarrow \text{HL} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{q}|Y_i \Delta \hat{Y}_i| \tag{4}$$

$$\uparrow \text{SA} = \frac{1}{m}\sum_{i=1}^{m}[\![Y_i = \hat{Y}_i]\!] \tag{5}$$

On the other hand, FMeasure is a widely used evaluation metric in traditional classification, however, in MLC, three different approaches are usually used to calculate it, such as Example-based FMeasure (ExF, Equation 2), Micro FMeasure (MiF, Equation 6), and Macro FMeasure (MaF, Equation 7). In these equations, tp_i, fp_i, and fn_i stands for the number of *true positives*, *false positives*, and *false negatives* of the i-th label, respectively. ExF is calculated for each instance, so it captures compound dependencies among labels. MiF first joins the confusion matrices of all labels and then calculates the metric, thus giving more weight to more frequent labels. MaF calculates the metric for each label and then averages their values, thus giving the same weight to each of the labels. Thus, there are three different approaches to calculate FMeasure, each treating in a different way the relationship and imbalance issues.

$$\uparrow \text{MiF} = \frac{\sum_{i=1}^{q} 2 \cdot tp_i}{\sum_{i=1}^{q} 2 \cdot tp_i + \sum_{i=1}^{q} fp_i + \sum_{i=1}^{q} fn_i} \tag{6}$$

$$\uparrow \text{MaF} = \frac{1}{q}\sum_{i=1}^{q}\frac{2 \cdot tp_i}{2 \cdot tp_i + fp_i + fn_i} \tag{7}$$

4.3 Experimental settings

The goal of the experimental studies is to compare the performance of the proposed CCEA with other state-of-the-art EMLCs. Therefore, we selected the EMLCs with better performance [12], as well as EME, which also uses an EA to build EMLCs. The datasets were partitioned using random 5-fold cross-validation procedure, and all methods were executed using 6 different seeds; then, the results were averaged over 30 different runs. The experiments were performed on a machine with Rocks cluster O.S., Intel Xeon E5645 Processor (6 $\times$ 2.40 GHz) and 64 GB RAM.

The default parameters proposed by their authors are used for each method. Unless otherwise specified, EMLCs use $n = 10$ members in the ensemble and LP with C4.5 decision tree [17] as the single-label classifier. Both EBR and ECC use sampling with replacement of the original training dataset at each member. EPS uses sample without replacement. RAkEL uses $n = 2q$ members and $k = 3$ labels. Finally, EME was run using 50 individuals in all cases, while the number of generations ranges from 110 to 300 depending on the dimensionality of the label space. As in RAkEL, EME uses $n = 2q$ members and $k = 3$.

In CCEA we use $k = 3$, just as in EME and RAkEL. However, in order to have on average 10 votes for each label (as in EBR, ECC, and EPS), the ensemble is composed by $n = \|3.33q\|$ members. Further, the number of individuals of the whole population is $2n$, evenly distributed among subpopulations. For the selection of the rest of parameters, a preliminary study was performed, which is available in additional material[7]. We fixed the maximum number of generations $n_g = 50$ in all cases and the number of generations between communications of subpopulations to $n_{gc} = 5$, so subpopulations have some generations to evolve by themselves until they communicate. Crossover and mutation probabilities were fixed to $p_c = 0.7$ and $p_m = 0.2$ in smaller datasets ($q < 30$), and $p_c = 0.7$ and $p_m = 0.1$ for bigger datasets ($q \geq 30$). The number of subpopulations ($n_s \in \{3, 4, 5\}$) and value of β in the ensemble selection and subpopulations update ($\beta \in \{0.25, 0.5, 0.75\}$) were selected by experimentation. In all cases, each subpopulation uses a random subset of 75% of the instances, sampled without replacement.

To determine if significant performance differences existed among the different EMLCs, we use Skillings-Mack's [2] and Bonferroni-Dunn's statistical tests [3]. Skillings-Mack's test is used to determine if the performance of the algorithms is statistically different. It is similar to Friedman's test, but it could be used with missing values. Further, Bonferroni-Dunn's test is used to perform pairwise comparisons with the control algorithm in each case. In order to perform comparisons without specifying a significance level and provide more statistical information, the adjusted p-values were used [5].

5 RESULTS AND DISCUSSION

Due to space constraints, in this section we present a summary of the experimental results; full results are available in additional material[7].

The results are summarized in Table 2, showing the average ranking for each of the EMLCs. For each dataset-metric pair, the best method is given a ranking of 1, the second best a ranking of 2, etc. The final ranking for each metric is calculated as the average value of each method over all datasets. Note that in the two most complex datasets, EME was not able to build a model within 2 days of execution. In these cases, the missing value is replaced by the average value of ranking among the rest of algorithms for the given data, as proposed in [2]. The last column shows the meta-ranking, calculated as the average value of ranking for each method over all metrics.

Table 2: Average rankings.

	HL	SA	ExF	MiF	MaF	*Meta-rank*
CCEA	3.54	3.12	**2.27**	**1.88**	1.96	**2.55**
EME	4.00	3.96	4.08	3.85	3.00	3.78
ECC	2.50	**2.69**	2.88	3.08	3.96	3.02
EBR	**1.69**	4.15	4.69	4.65	4.92	4.02
RAkEL	5.08	3.81	2.85	2.81	**1.85**	3.28
EPS	4.19	3.27	4.23	4.73	5.31	4.35

As can be seen, the CCEA is the best ranked method overall, being the best in two of the metrics, while EBR, ECC, and RAkEL are the best in one metric each. Besides, except for MaF, in all cases CCEA obtains a better average ranking than both RAkEL and EME, which are based on learning small k-labelsets. Further, note that in SA and MaF metrics the CCEA is the second best, and third in HL. RAkEL achieves the worst performance in HL; EBR is always between the two last positions in all metrics except for HL; and ECC is fourth in MaF.

Skillings-Mack's test results are shown in Table 3. It determines that for all the metrics but SA, the performance of the EMLCs is statistically different, so Bonferroni-Dunn's post-hoc test is also performed for those 4 metrics. Table 4 shows the adjusted p-values of the comparison of the EMLCs using the control algorithm in each case, which is the best method for a given metric. For each metric, the control algorithm is indicated using "-", and those methods which performance is statistically different to the control algorithm at 95% confidence are shown in bold.

[7] Additional material available at http://www.uco.es/kdis/CCEA

Table 3: Results of Skillings-Mack's test.

	Skillings-Mack statistic	p-value
HL	27.80	**3.98E-5**
SA	5.90	3.16E-1
ExF	17.12	**4.28E-3**
MiF	23.15	**3.15E-4**
MaF	40.33	**1.28E-7**

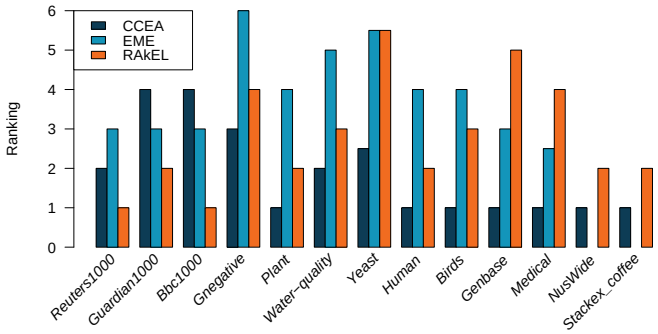
Table 4: Results of Bonferroni-Dunn's test.

	HL	ExF	MiF	MaF
CCEA	5.94E-02	-	-	4.38E+00
EME	**8.31E-03**	6.88E-02	**3.76E-02**	5.79E-01
ECC	1.36E+00	2.01E+00	5.21E-01	**1.97E-02**
EBR	-	**4.80E-03**	**8.04E-04**	**1.38E-04**
RAkEL	**2.00E-05**	2.16E+00	1.04E+00	-
EPS	**3.29E-03**	**3.76E-02**	**5.25E-04**	**1.20E-05**

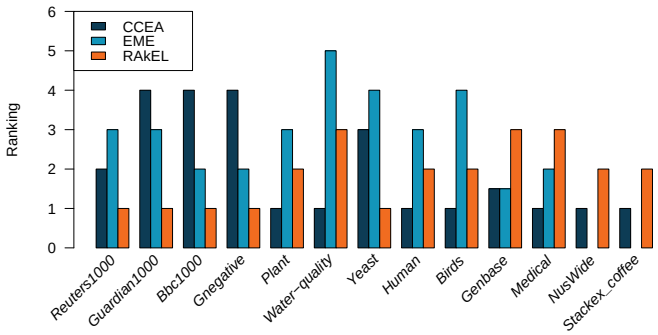
From the results we can reach several conclusions. First, the proposed CCEA is able to outperform EME. The fact of evolving individuals as separate members of the ensemble instead of using the entire ensemble allows the CCEA to build an ensemble with more promising members, while considering both performance and diversity. On the other hand, the use of different subpopulations, each using a different sample of the original training dataset, introduces the necessary diversity in the EMLC.

Second, we have shown that the CCEA had statistically better and more consistent performance than state-of-the-art EMLCs. It has the best average ranking among all metrics, being the best in two of them, and also is the only method that does not perform statistically worse than the control algorithm in any of the cases. EPS performs statistically worse than the control method in all cases. EBR, which is the best method in HL, performs statistically worse than the control method in the rest of the metrics. ECC and RAkEL, which achieve good performance in several metrics, perform statistically worse than the control algorithm in one metric each at 95% confidence. The CCEA is the only algorithm whose performance is statistically the same than the control algorithm in all cases, which shows its consistency.

Finally, a large number of labels means that in EMLCs where each member is focused on a small k-labelset (such as CCEA, EME, and RAkEL), the possible number of different k-labelsets and the possible number of combinations of members into an ensemble, grows exponentially. Thus, in order to study the performance of the CCEA in regard to dimensionality of the output space, in Figure 6 we show the ranking of CCEA, EME, and RAkEL in each dataset for two of the metrics, the MiF and MaF. Since rankings are presented in both figures, the lower the value, the better the performance. In both figures the datasets are ordered by ascending number of labels. We can see that as the number of labels increases, the CCEA obtains an optimal combination of members for the ensemble. This means that the CCEA is suitable for datasets with a large output space. Since for the two most complex datasets EME did not finish its execution, its ranking is not shown in the figures.



(a) MiF



(b) MaF

Figure 6: Ranking of CCEA, EME, and RAkEL for each dataset. The ranking of EME in the two most complex datasets is not shown since it did not finish its execution.

6 CONCLUSIONS

In this paper we propose a cooperative coevolutionary algorithm to build EMLCs. In CCEA algorithm, each individual of the population is a possible member of the ensemble. Several subpopulations exists simultaneously, where each of them use a different subset of the training data to build multi-label classifiers, providing more diversity to the ensemble. The evaluation of the individuals is performed over the full training dataset, thus allowing to evaluate individuals over some unseen instances as well as to better know how they will perform when combined into the ensemble. Further, each n_{gc} generations, subpopulations communicate among them, not only building an EMLC using individuals from all subpopulations, but also sharing information between them, thanks to the used genetic operators.

The experimental study carried out using 13 multi-label datasets and 5 evaluation metrics demonstrated that the proposed CCEA has a statistically better and more consistent performance than state-of-the-art EMLCs. The CCEA is not only the method with better average ranking among all metrics but also it is the only one which does not perform statistically worse than the control algorithm in any of the cases.

In the future, we will work on other ways to communicate between the subpopulations, as well as define other criteria to increase the diversity of each subpopulation.

ACKNOWLEDGEMENTS

This research was supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund, project TIN2017-83445-P. This research was also supported by the Spanish Ministry of Education under FPU Grant FPU15/02948.

REFERENCES

[1] Yaxin Bi, 'The impact of diversity on the accuracy of evidential classifier ensembles', *International Journal of Approximate Reasoning*, **53**(4), 584 – 607, (2012).

[2] M. Chatfield and A. Mander, 'The skillings–mack test (friedman test when there are missing data)', *The Stata journal*, **9**(2), 299–305, (2009).

[3] Olive Jean Dunn, 'Multiple comparisons among means', *Journal of the American Statistical Association*, **56**(293), 52–64, (1961).

[4] Agoston E Eiben, James E Smith, et al., *Introduction to evolutionary computing*, volume 53, Springer, 2003.

[5] Salvador Garcia and Francisco Herrera, 'An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons', *Journal of Machine Learning Research*, **9**(Dec), 2677–2694, (2008).

[6] Ouadie Gharroudi, Haytham Elghazel, and Alex Aussem, 'Ensemble Multi-label Classification: A Comparative Study on Threshold Selection and Voting Methods', in *IEEE International Conference on Tools with Artificial Intelligence,*, pp. 377–384, (2015).

[7] Eva Gibaja and Sebastián Ventura, 'Multi-label learning: a review of the state of the art and ongoing research', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **4**(6), 411–444, (2014).

[8] Eva Gibaja and Sebastian Ventura, 'A tutorial on multilabel learning', *ACM Computing Surveys*, **47**(3), (2015).

[9] Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Džeroski, 'Ensembles of multi-objective decision trees', in *European conference on machine learning*, pp. 624–631, (2007).

[10] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera, 'An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics', *Information sciences*, **250**, 113–141, (2013).

[11] E Loza and J Fürnkranz, 'Efficient multilabel classification algorithms for large-scale problems in the legal domain', in *Semantic Processing of Legal Texts*, volume 6036, pp. 192–215, (2010).

[12] Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura, 'Review of ensembles of multi-label classifiers: Models, experimental study and prospects', *Information Fusion*, **44**, 33 – 45, (2018).

[13] Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura, 'An evolutionary approach to build ensembles of multi-label classifiers', *Information Fusion*, **50**, 168–180, (2019).

[14] Jose M. Moyano, Eva L. Gibaja, and Sebastián Ventura, 'An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains', in *2017 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2015–2021, (2017).

[15] Jose M. Moyano, Eva Lucrecia Gibaja, and Sebastián Ventura, 'MLDA: A tool for analyzing multi-label datasets', *Knowledge-Based Systems*, **121**, 1–3, (2017).

[16] Mitchell A Potter and Kenneth A De Jong, 'A cooperative coevolutionary approach to function optimization', in *International Conference on Parallel Problem Solving from Nature*, pp. 249–257, (1994).

[17] J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., 1993.

[18] J Read, 'A pruned problem transformation method for multi-label classification', in *Proceedings of the NZ Computer Science Research Student Conference*, pp. 143–150, (2008).

[19] Jesse Read, Bernhard Pfahringer, and Geoff Holmes, 'Multi-label classification using ensembles of pruned sets', in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pp. 995–1000. IEEE, (2008).

[20] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank, 'Classifier chains for multi-label classification', *Machine Learning*, **85**(3), 335–359, (2011).

[21] H. Shao, G.Z. Li, G.P. Liu, and Y.Q. Wang, 'Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine', *Science China Information Sciences*, **56**(5), 1–13, (2013).

[22] G. Tsoumakas and I. Katakis, 'Multi-label classification: An overview', *International Journal of Data Warehousing and Mining*, **3**(3), 1–13, (2007).

[23] G Tsoumakas, I Katakis, and I Vlahavas, 'Effective and efficient multilabel classification in domains with large number of labels', in *Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, pp. 53–59, (2008).

[24] G Tsoumakas, I Katakis, and I Vlahavas, *Data Mining and Knowledge Discovery Handbook, Part 6*, chapter Mining Multi-label Data, 667–685, Springer, 2010.

[25] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas, 'Random k-labelsets for multi-label classification', *IEEE Transactions on Knowledge and Data Engineering*, **23**(7), 1079–1089, (2011).

[26] Grigorios Tsoumakas, Ioannis Partalas, and Ioannis Vlahavas, 'A taxonomy and short review of ensemble selection', in *Workshop on Supervised and Unsupervised Ensemble Methods and Their Applications*, pp. 1–6, (2008).

[27] J. Xu, 'Fast multi-label core vector machine', *Pattern Recognition*, **46**(3), 885–898, (2013).

7.3 Tree-shaped ensemble of multi-label classifiers using grammar-guided genetic programming

Title	Tree-shaped ensemble of multi-label classifiers using grammar-guided genetic programming
Authors	Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastián Ventura
Conference	IEEE Congress on Evolutionary Computation (IEEE CEC 2020)
Status	Accepted paper
Year	2020

Tree-Shaped Ensemble of Multi-Label Classifiers using Grammar-Guided Genetic Programming

Jose M. Moyano*, Eva L. Gibaja*, Krzysztof J. Cios†‡ and Sebastián Ventura*
*Dept. of Computer Science and Numerical Analysis
University of Córdoba, Córdoba, Spain
Email: jmoyano@uco.es, egibaja@uco.es, sventura@uco.es
†Dept. of Computer Science
Virginia Commonwealth University, Richmond, VA, USA
Email: kcios@vcu.edu
‡ Polish Academy of Sciences, Poland

***Abstract*—Multi-label classification paradigm has had a growing interest because of the emergence of a large number of classification problems where each of the instances of the data can be associated with several output labels simultaneously. Several ensemble methods were proposed to solve the multi-label classification problem. However, most of them simply create diversity in the ensemble by following a random procedure and give the same importance to all members. In this paper, we propose a Grammar-Guided Genetic Programming algorithm to build ensembles of multi-label classifiers. Given a pool of multi-label classifiers, each of them modeling dependencies among a subset of *k* labels, they are combined into a tree-shaped ensemble. At each node of the tree, predictions of its children nodes are combined, while each leaf represents a classifier from the pool. We propose two configurations for the method: using a fixed value of *k* for all classifiers in the pool, or using a variable value of *k* for each classifier, thus being able to capture relationships among groups of labels of different size in the ensemble. The experiments performed over sixteen multi-label dataset and using five evaluation metrics demonstrated that our method performs significantly better than the state-of-the-art ensembles of multi-label classifiers.**

***Index Terms*—Genetic programming, Multi-label classification, Ensemble learning**

I. Introduction

In recent years, classification problems where each of the instances of the data may belong to several classes/output labels simultaneously associated with it, are increasingly frequent. For example, in medical diagnosis systems, patients may have more than one disease, or complications, at the same time. Traditional classification methods are only able to deal with one class per instance. Thus, the Multi-Label Classification (MLC) paradigm emerged for addressing these situations [1]. MLC has been successfully applied to many real-world problems in addition to medical diagnosis [2], such as biology [3] and multimedia categorization [4].

Dealing with several output labels at the same time leads to emergence of new challenges such as modeling the compound dependencies among the labels, imbalance, and high dimensionality of the output space. Although a wide range of MLC methods has been proposed [1], we focus here our attention on Ensembles of Multi-Label Classifiers (EMLCs), which have been shown to outperform the base methods [5]–[7]. Ensemble classifiers are methods that combine predictions of several base classifiers, aiming to improve the overall generalization ability of each. Selection of classifiers to combine, however, is not trivial as they should not only be accurate but also diverse [8], [9].

Although the EMLCs outperform their base methods, they usually combine classifiers where the diversity is created by following any random procedure (such as randomly selecting instances or labels at each member of the ensemble), and the same weight or importance is given to all base classifiers. In this paper, we propose a Grammar-Guided Genetic Programming (G3P) algorithm able to generate EMLCs. G3P, which is an extension of Genetic Programming (GP), is an evolutionary learning technique that uses syntax trees to represent the individuals, and also a grammar to guide the learning process, such as the creation of initial individuals [10]. Using G3P a tree-shaped ensemble is obtained; at each node of the tree the predictions of children nodes are combined, while the leaves are the base multi-label classifiers. The use of G3P makes the selection of members of the ensemble more flexible, allowing to adapt to each particular problem, as well as to obtain an optimal structure of the ensemble.

In our method each of the base classifiers of the ensemble focuses only on a subset of k labels, also known as k-labelset. In this way, each member is able to consider the relationship among the labels, while drastically reducing the imbalance and high-dimensionality of the output space. In addition, our method is not only able to deal with fixed k for all base classifiers, but also is able to use different values of k in each base classifier, to capture the relationship among subsets of labels of different size. The experimental study using 16 datasets and five evaluation metrics, demonstrated that our method obtains significantly better performance than state-of-the-art MLC methods.

The rest of the paper is organized as follows. Section II provides background about MLC and G3P; Section III describes the G3P-based method for building the EMLCs; Section IV introduces the experimental study; Section V presents and discusses the results; and Section VI ends with conclusions.

II. Background

In this section, we first describe MLC and state-of-the-art MLC methods, and then we introduce the G3P framework.

A. Multi-Label Classification

Let $\mathcal{X} = X_1 \times \cdots \times X_d$ be the d-dimensional input space, and $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ the output space composed by $q > 1$ labels. Let $\mathcal{D}$ be a multi-label dataset composed of m instances, as $\mathcal{D} = \{(\mathbf{x}_i, Y_i) | 1 \leq i \leq m\}$, where each multi-label instance is a pair composed by an input feature vector $\mathbf{x} \in \mathcal{X}$ and a set of relevant labels $Y \subseteq \mathcal{Y}$ associated with it [1]. The goal of MLC is to construct a model able to provide a set of predicted relevant labels $\hat{Y}$ for an unknown instance $\mathbf{x}$.

MLC algorithms are categorized into three groups: I) problem transformation, which transform the multi-label problem into one or several single-label problems; II) algorithm adaptation, which directly adapt traditional classification methods to be able to deal with multi-label data without transforming it; and III) EMLCs, defined as a set of n multi-label classifiers, each of them providing prediction for all or part of the labels [7]. Finally, the prediction of all n classifiers are combined, usually by majority voting, although many other combining methods can be used [11].

Ensemble of Binary Relevances (EBR) algorithm [5] combines the predictions of n Binary Relevance (BR) methods [12]. As BR creates an independent binary model for each label, EBR is not able to model the relationship among the labels. Also, diversity of classifiers in EBR is obtained by simply randomly selecting a subset of the instances for each of the members, which is a weakness.

Ensemble of Classifier Chains (ECC) [5] combines the predictions of n Classifier Chain (CC) methods. CC also creates binary models for each label, but they are not independent as in BR as they are chained in such a way that predictions of previous labels in the chain are introduced as additional input features in the subsequent binary classifiers. Therefore, ECC is able to model some of the relationship among labels. The diversity in ECC is obtained not only by selecting random subsets of the instances but also using different random chains at each member.

Ensemble of Pruned Sets (EPS) algorithm [13] combines the predictions of n Pruned Sets (PS) methods. PS follows the Label Powerset (LP) [14] approach, transforming the multi-label problem into a multi-class problem, where different combinations of labels (a.k.a. labelsets) are considered as different classes; then, PS prunes those instances associated with very infrequent classes, leading to less imbalanced problems. In this way, EPS is able to model the relationship among all labels; however, although it prunes infrequent labelsets, the resulting multi-class problem still tends to be imbalanced with a high number of classes, resulting in a still complex problem. As for the diversity, EPS selects random subset of instances at each member of the ensemble.

RAndom *k*-labELsets (RA*k*EL) [6] builds an ensemble of LP methods, but in this case each member of the ensemble focuses on a small subset of k labels, being k fixed for all members. Therefore, each member of the ensemble is able to deal with the relationships among k labels, leading to less imbalanced and lower dimensional problems than if all labels were considered at the same time. RA*k*EL selects the k-labelsets randomly to generate diversity in the ensemble.

Random Forest of Predictive Clustering Trees (RF-PCT) [15] builds an ensemble of Predictive Clustering Trees (PCTs) [16]. Each member uses a random selection of training instances, and at each node of the tree selects the best feature from a random subset of attributes.

In all described methods, the diversity among the members is generated following a random process, and the combination of labels is performed by majority voting, with the same weight given to each member of the ensemble.

B. Grammar-Guided Genetic Programming (G3P)

GP is an evolutionary and very flexible heuristic technique which allows the use of very complex individual representations. G3P is an extension of GP, where a free-context grammar is used to generate the individuals. Each individual is represented as a syntax tree, where internal or non-terminal nodes correspond to functions taking their children as arguments, and leaves or terminal nodes correspond to variables and constants. The use of the grammar provides ability of applying certain constraints at each node of the tree, such as the number or type of the child nodes, and also ensures that all generated individuals represent a valid solution [17].

G3P has been widely used in the literature on a large number of different problems, such as bankruptcy prediction [18], predicting student performance [19], and discovery of subgroups within a population [20], as well as it has been proven to work well on high-dimensional scenarios [21]. However, there are not many multi-label classification methods that are based on G3P. In [22] a G3P algorithm to build a rule-based multi-label classifier was proposed. In [23], an algorithm that automatically selects the most appropriate multi-label classifier is proposed using G3P. Nonetheless, to the best of our knowledge, no studies exist in applying either GP or G3P to the construction of EMLCs.

III. G3P-kEMLC

In this section we present the proposed method, called G3P-kEMLC. First, the main steps of the algorithm are presented, and then, the individuals, fitness function, and genetic operators are described.

A. Algorithmic strategy

The main steps of the G3P-kEMLC algorithm are shown in Fig. 1. First, a pool of n multi-label classifiers $\text{MLC}_1, \text{MLC}_2, \ldots, \text{MLC}_n$, where each is focused on predicting k labels, is created. Although any multi-label classifier could be used, we will use LP, in order to model the relationship among all k labels at the same time [6]. Unlike other methods, such as RA*k*EL, G3P-kEMLC is able to handle multi-label classifiers using different values of k. Therefore, two parameters k_{min} and k_{max} are given, so that each classifier

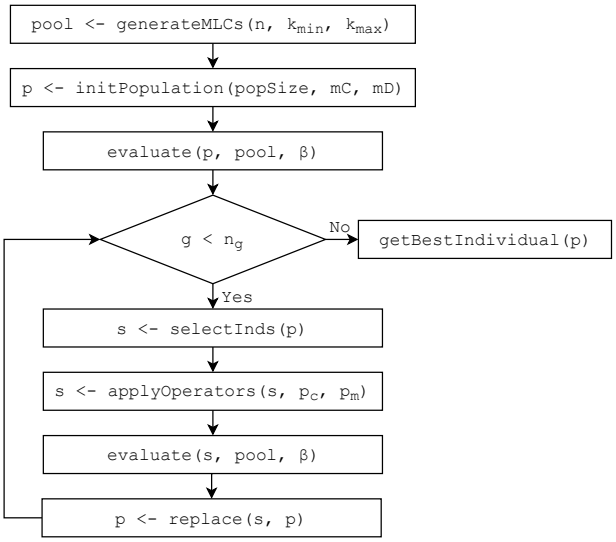


Fig. 1. Main steps of G3P-kEMLC.

will focus on a subset of k labels, being k a random value in the range $[k_{\text{min}},\ k_{\text{max}}]$. Note that repeated k-labelsets are not allowed, so if the generated random k-labelset is currently present in the pool, it is discarded and a new one is created. To increase diversity of the base classifiers, each of them is built over a random subset of the instances. Although building the base classifiers over random subsets of instances and labels, the difference between G3P-kEMLC and other EMLCs is that G3P-kEMLC will look for an optimal structure of the ensemble by means of the evolutionary procedure, instead of just giving the same weight to all base classifiers in the combination of predictions. In our tree-shaped ensemble, classifiers that are placed at a shallower depth have more importance in the final prediction than classifiers that are deeper in the tree.

Then, the initial population p of $popSize$ individuals is generated by using the grammar (see Section III-B). Note that mC and mD parameters indicate the maximum number of children of each node of the tree, and the maximum allowed depth of the tree, respectively. Once initial individuals are created, they are evaluated (see Section III-C).

Until the maximum number of generations n_g is reached, individuals are selected by tournament selection, genetic operators are applied (see Section III-D), and new individuals are evaluated. For the replacement of the population, the population at next generation is formed by all new children. However, if the best parent is better than the best child, it replaces the worst child, thus maintaining elitism. Once the maximum number of generations is reached, the best individual is returned as the optimal ensemble.

B. Individuals

Each individual in G3P-kEMLC is represented as a string encoding a tree-shaped ensemble. As non-terminal nodes, only one function called *Comb* is used, which combines the

```
<S>: <Comb>
<Comb>: <start> (<Comb>|<MLC>){2, mC} <end>
<start>: '('
<end>: ')'
<MLC>: MLC_1 | MLC_2 | ... | MLC_n
```

Fig. 2. Free-context grammar.

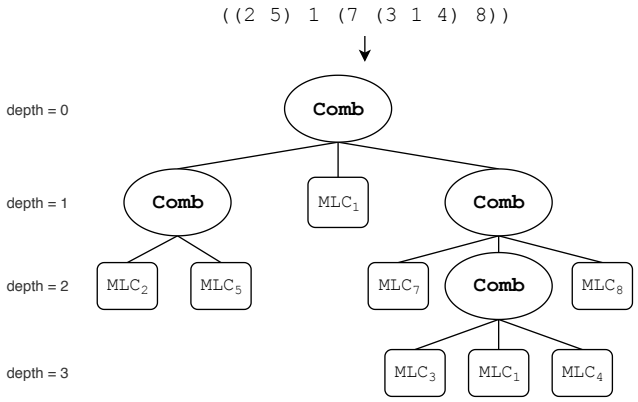


Fig. 3. Individual as string and its corresponding tree.

predictions of children nodes. For each label λ_l that is included in any of the children of the *Comb* node, it computes the ratio of positive predictions among all the children; if it is greater or equal than a given threshold t (as default $t = 0.5$ is used), the combined prediction for λ_l is positive, and negative otherwise. On the other hand, as terminal nodes, we use the previously generated pool of multi-label classifiers, represented as integers from 1 to n. Each of the multi-label classifiers gives prediction for the labels in their own k-labelsets. Note that, since each leaf is focused on a subset of k labels (and k could be different for each leaf), the number of labels considered at each *Comb* node can be different.

In Fig. 2 the grammar used to build individuals is shown. The initial node `S` is always replaced by a *Comb* node. Each *Comb* node starts and ends with '(' and ')' characters respectively, to represent (in the string) the hierarchy of the nodes. Each *Comb* node may contain from 2 to mC children nodes, each of them being either another *Comb* node or a *MLC* (leaf). The number of children is randomly selected in the given range $[2, mC]$ at each node, so each internal node could have a different number of children. However, the maximum depth (mD) of the tree is controlled in such a way that if the depth of the current *Comb* node is equal to $mD - 1$, only *MLC* nodes can be selected as children of this *Comb*. In Fig. 3 an example of an individual obtained using the grammar is shown, both as a string and as its corresponding tree.

The use of the grammar is helpful in two main aspects. On the one hand, it allows to determine the number of child nodes at each *Comb* node, as well as if these children are either other *Comb* nodes or leaves. On the other hand, if different types of combiner nodes for the predictions were used (as

we propose for future work), the use of the grammar would become essential, since it would select the appropriate type and/or number of child nodes depending on different type of combiner node.

C. Evaluation

For evaluation of the individuals, we use a fitness function (Eq. 1) which differentiates between incomplete and complete trees. Hereafter, it is indicated with ↑ and ↓ if metrics are maximized or minimized, respectively. L_t is the set of labels that the tree is considering among all the leaves. We define complete trees as those that include at least one vote for each label in the dataset ($|L_t| = l$), while incomplete trees are not able to give prediction for all labels ($|L_t| < l$).

$$\uparrow fitness = \begin{cases} -\left(l - |L_t|\right)/l & \text{if } |L_t| < l \\ \beta \cdot \text{ExF} + (1-\beta) \cdot \text{MaF} & \text{if } |L_t| = l \end{cases} \quad (1)$$

If the individual represents an incomplete tree, its fitness is a negative value, being closer to zero as the number of labels that are not considered is lower. We aim to remove incomplete individuals in the population, but in case when several incomplete individuals are chosen to compete with each other in the selection procedure, the one that is closer to be a complete tree is selected.

On the other hand, if the individual is a complete tree, first the predictions for the whole training set are obtained by combining predictions of internal nodes, until the final prediction of the ensemble is obtained at the root node. Then, the fitness function, composed of two terms, is computed. FMeasure is a robust evaluation metric that has been widely used to evaluate models in cases where the output space is imbalanced [24]. In MLC, several approaches are defined to compute the FMeasure, such as Example-based FMeasure (ExF) and Macro-averaged FMeasure (MaF) [25]. ExF (Eq. 2) computes the FMeasure of each instance as a whole, thus capturing the relationships among the labels in its calculation. On the other hand, MaF (Eq. 3) computes the FMeasure for each label independently and averages it by the number of labels, thus giving the same importance to all labels in its calculation. Note that tp_l, fp_l, and fn_l stand for the number of *true positives*, *false positives*, and *false negatives* of the l-th label, respectively.

$$\uparrow \text{ExF} = \frac{1}{m} \sum_{i=1}^{m} \frac{2|\hat{Y}_i \cap Y_i|}{|\hat{Y}_i| \cup |Y_i|} \quad (2)$$

$$\uparrow \text{MaF} = \frac{1}{q} \sum_{l=1}^{q} \frac{2 \cdot tp_l}{2 \cdot tp_l + fp_l + fn_l} \quad (3)$$

Therefore, using a combination of both ExF and MaF, we not only consider the relationship among labels when calculating the FMeasure, but also ensure that minority labels are also considered. Further, the fact of calculating the fitness function over the whole training set, while each base classifier is built over a subset of the same data, also offers an approximation of how each of them performs on unseen data.

D. Genetic operators

Each individual of the population is selected for crossover and mutation operators based on probabilities p_c and p_m respectively. Fig. 4 illustrates use of both operators.

1) Crossover operator: Given two parents, the crossover operator creates a new individual as follows: I) a random subtree st_1, not including the whole tree, is selected from the first parent; II) a random subtree st_2, including the possibility to select the whole tree, is selected from the second parent; III) if the maximum depth of st_2 is greater than the maximum allowed depth of the tree minus the depth of the root node of st_1, step II is repeated again but selecting a random subtree of st_2; IV) st_2 replaces st_1 in the first parent, obtaining the child individual. Crossed individuals include genetic material of both parents, and because of step III they are always feasible, as we ensure not to exceed the maximum allowed depth of the tree.

Regarding the example in Fig. 4a, consider that the subtree whose root is the shaded node of the first parent is selected as st_1. Then, the node marked with a dotted line in the second parent is first selected as st_2. However, considering $mD = 3$, if st_2 replaces st_1, the maximum depth constraint would not be met in the generated child; therefore, a new random subtree below this node is then selected as st_2 (shaded node of the second parent). Finally, the first child is created by replacing st_1 with st_2.

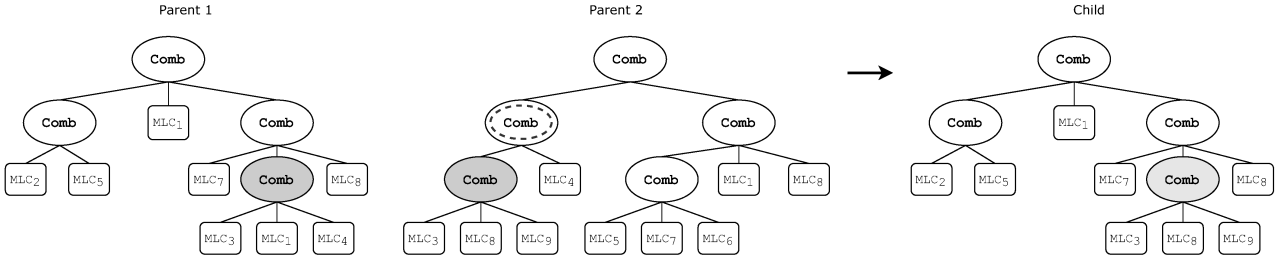
Given this crossover operator, just one child is obtained by each pair of parents. Proposing an operator where two children were obtained, for example swapping subtrees of the parents, would make more difficult the selection of these subtrees in such a way that they both fit in the other parent and do not break the maximum depth restriction in any of them. Therefore, the second child is obtained by following the same procedure but swapping the roles of each of the parents.

2) Mutation operator: For the mutation operator, the steps are: I) a random subtree st, not including the whole tree, is selected; II) a subtree is created following the grammar, but using as maximum depth the maximum allowed depth minus the depth of the root node of st; III) the generated tree replaces st. Therefore, a subtree of the individual is replaced by a random subtree, ensuring that it is feasible thanks to the use of the grammar and control of the maximum depth.

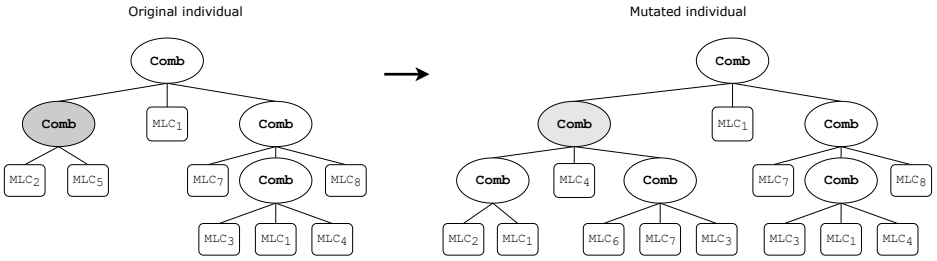
According to Fig. 4b, assume that the shaded node is randomly selected as root of the subtree to mutate. Then, a random new subtree is created following the grammar to replace it, creating the mutated individual.

IV. Experimental Study

In this section, first the datasets and evaluation metrics used in the experiments are described, and then, the configuration of both G3P-kEMLC and the state-of-the-art MLC methods are presented.



(a) Crossover



(b) Mutation

Fig. 4. Genetic operators.

TABLE I
DATASETS AND CHARACTERISTICS. THE DATASETS ARE ORDERED BY THE NUMBER OF LABELS.

Dataset	m	d	q	*card*	*avgIR*	*rDep*
Emotions	593	72	6	1.868	1.478	0.933
Reuters1000	294	1000	6	1.126	1.789	0.667
Guardian1000	302	1000	6	1.126	1.773	0.667
Bbc1000	352	1000	6	1.125	1.718	0.733
3s-inter3000	169	3000	6	1.142	1.766	0.400
GnegativePseAAC	1392	1717	8	1.046	18.448	0.536
PlantPseAAC	978	440	12	1.079	6.690	0.318
Water-quality	1060	16	14	5.073	1.767	0.473
Yeast	2417	103	14	4.237	7.197	0.670
HumanPseAAC	3106	440	14	1.185	15.289	0.418
Birds	645	260	19	1.014	5.407	0.123
Genbase	662	1186	27	1.252	37.315	0.157
Medical	978	1449	45	1.245	89.501	0.039
NusWide[2]	2696	128	81	1.863	89.130	0.087
Stackex coffee	225	1763	123	1.987	27.241	0.017
CAL500	502	68	174	26.044	20.578	0.192

A. Datasets

A selection of 16 multi-label datasets from the KDIS repository[1], covering a wide range of characteristics, were used in the experiments and are shown in Table I. The number of instances (m), attributes (d), and labels (q) of each dataset are shown, as well as the cardinality or average number of labels associated with each instance (*card*), the average imbalance ratio (*avgIR*), and the ratio of dependent label pairs (*rDep*) [26].

B. Evaluation metrics

Five evaluation metrics are used to assess the performance of the multi-label methods [25]. The Adjusted Hamming loss (AHL) [27] was proposed because of the drawbacks of the Hamming loss, which tends to be zero in cases with a large number of labels but low cardinality. AHL, defined in Eq. 4, computes the ratio of misclassified labels divided by the number of positive labels (in both true and predicted sets), and averages it by the total number of instances. Note that Δ is the symmetric difference between two binary sets.

$$\downarrow \text{AHL} = \frac{1}{m}\sum_{i=1}^{m}\frac{|\hat{Y}_i \Delta Y_i|}{|\hat{Y}_i \cup Y_i|} \tag{4}$$

Subset accuracy (SA), Eq. 5, is a strict metric which evaluates the number of instances where the set of predicted labels exactly matches the set of true labels. Note that $[\![\pi]\!]$ returns 1 if predicate π is true, and 0 otherwise.

$$\uparrow \text{SA} = \frac{1}{m}\sum_{i=1}^{m}[\![\hat{Y}_i = Y_i]\!] \tag{5}$$

Also, we use different versions of the FMeasure, namely Example-based FMeasure (ExF; Eq. 2), Micro-averaged FMeasure (MiF; Eq. 6), and Macro-averaged FMeasure (MaF; Eq. 3). While ExF captures the relationship among the labels in its calculation, MiF and MaF give different weight to the labels; MiF is biased by more frequent labels, while MaF gives the same importance to all of them.

$$\uparrow \text{MiF} = \frac{\sum_{l=1}^{q} 2 \cdot tp_l}{\sum_{l=1}^{q} 2 \cdot tp_l + \sum_{l=1}^{q} fp_l + \sum_{l=1}^{q} fn_l} \tag{6}$$

[1]http://www.uco.es/kdis/mllresources

[2]In order to execute it in reasonable time, a random selection of instances of NusWide cVLAD+ dataset was performed.

C. Methods and configurations

G3P-kEMLC has been built using JCLEC [28], Mulan [29], and Weka [30] libraries, and the code is publicly available in a GitHub repository[3].

In the experiments, we use two different configurations of G3P-kEMLC: I) with a fixed value of $k = 3$ for all base classifiers; and II) with a variable value of k for each of them. For the second configuration, the size of the k-labelset of each multi-label classifier is in the range $[3, q/2]$. In this way, we observe how the method works both using fixed k value as in RAkEL, and also by considering the relationships among a large number of labels. Regarding the size of the pool of classifiers, Eq. 7 indicates how to calculate the number of classifiers needed to have, on average, $\overline{v}$ votes for each label in the pool. For n we also use two different configurations: I) n for $\overline{v} = 10$; and II) n for $\overline{v} = 20$. Note that in our approach, the fact of using a large number of base classifiers in the pool does not mean that all of them will be included in the ensemble, since the G3P algorithm selects the most suitable models. For each dataset, the results of the best of these two configurations are reported; the best configuration is the one with better ranking among all five evaluation metrics.

$$n = \frac{\overline{v}}{(k_{min} + k_{max})/2} \cdot q \tag{7}$$

Concerning the size of the tree, we use the maximum number of children at each node $mC = 7$, while the maximum depth is fixed to $mD = 3$ for most cases. However, when using fixed $k = 3$, for those datasets with more than 50 labels, $mD = 4$ is used. Note that for example for NusWide dataset, which has 81 labels, the size of the pool for $\overline{v} = 20$ is $n = 540$, while a tree of $mD = 3$ and $mC = 7$ can have, as most, $7^3 = 343$ leaves, so it could not include all classifiers in the pool if it was necessary. Using $mD = 4$, more complex trees including at most $7^4 = 2401$ leaves can be created, which is enough for these cases.

For the selection of the rest of parameters of the algorithm, we performed a brief preliminary study. Due to space constraints, additional material including this study is available at the KDIS Group Webpage[4]. We use $popSize = 50$ individuals; $n_g = 150$ generations; probabilities $p_c = 0.7$ and $p_m = 0.2$ for genetic operators; $\beta = 0.5$ for the fitness function; and 75% of instances are sampled without replacement at each multi-label classifier. Finally, not only in G3P-kEMLC but also in other EMLCs, the C4.5 [31] decision tree algorithm is used as a single-label classifier (except for RF-PCT, which uses PCTs).

A comparison with RAkEL, which also uses subsets of k labels at each base classifier, shall demonstrate whether the fact of evolving an optimal structure for the ensemble instead of just giving the same weight to each classifier improves (or not) predictive performance. The recommended configuration for RAkEL is with $k = 3$ and $n = 2q$, so that each label has, on average, 6 votes. However, to perform a fair comparison, RAkEL was executed with $n = 2q$ (so $\overline{v} = 6$, as recommended), $n = 3.33q$ (meaning $\overline{v} = 10$, as in other EMLCs), and also with n calculated in such a way that $\overline{v}$ is the same as in the best configuration of G3P-kEMLC; we report the results of the best configuration. In this way, we aim to show that the performance of our method is not only biased by the number of base classifiers.

A second comparison involving state-of-the-art EMLCs was also performed. The best EMLCs according to the study in [7] were selected for the comparison. For all EMLCs, the default parameters proposed by their authors were used. EBR, ECC and RF-PCT use sampling with replacement of the original training dataset at each member, while EPS uses samples without replacement of 66% of the instances. Note that for CAL500 dataset, which has as many different labelsets as instances, EPS was executed without pruning the infrequent labelsets. Finally, all of them use $n = 10$.

In all cases, the datasets were partitioned using random 5-fold cross-validation, and all methods were executed using 6 different seeds for random numbers. Finally, the results were averaged over 30 runs.

V. Results and Discussion

In this section, the results of the experimental study are presented and discussed. First, analysis and comparison of G3P-kEMLC and RAkEL is performed; then, G3P-kEMLC is compared to other state-of-the-art EMLCs. Hereafter, the two versions of our proposed method are indicated as G3P-kEMLC-3 when $k = 3$ is used, and as G3P-kEMLC-V when a variable value of k in the range $[3, q/2]$ is used.

A study of the efficiency of G3P-kEMLC was also carried out; however, due to space constraints, the results of this study are available in the additional material, as well as detailed results of the experiments and also p-values of statistical tests.

A. G3P-kEMLC vs RAkEL

In this section, we compare the performance of G3P-kEMLC and RAkEL, as methods that use base classifiers focused on k-labelsets. In Table II, the average ranking values for each metric, computed for all datasets, are shown. For each pair dataset-metric, the method that performs best obtains a ranking of 1, the next a ranking of 2, and so on; the lower the value the better. We can see that in four of the metrics G3P-kEMLC-V has the best average ranking, while G3P-kEMLC-3 is the best in one. Further, both methods are ahead of RAkEL in all cases except in SA, where RAkEL has a better average ranking than G3P-kEMLC-3.

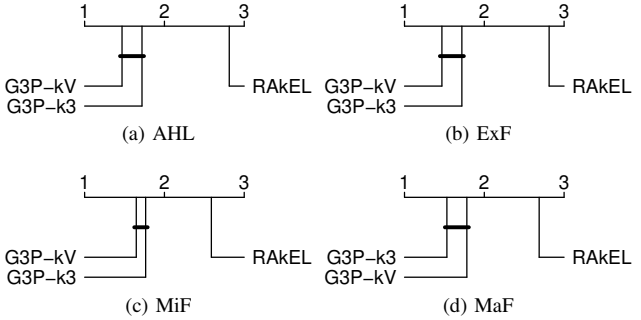
In order to determine if significant differences exist among algorithms, Friedman's [32] and Holm's [33] tests were performed, using a confidence value $\alpha = 0.05$. Friedman's test has shown that significant differences existed for all metrics except for SA. Further, Holm's test has shown that for the rest of the metrics, the performance of both configurations of G3P-kEMLC is statistically the same, and both are significantly better than RAkEL. Fig. 5 shows the critical diagrams for

[3]https://www.github.com/kdis-lab/G3P-kEMLC

[4]http://www.uco.es/kdis/G3P-kEMLC/

TABLE II
AVERAGE RANKINGS IN THE COMPARISON AMONG G3P-KEMLC AND RA*k*EL.

	G3P-kEMLC-3	G3P-kEMLC-V	RA*k*EL
AHL	1.72	**1.47**	2.81
SA	2.19	**1.81**	2.00
ExF	1.72	**1.47**	2.81
MiF	1.76	**1.65**	2.59
MaF	**1.53**	1.78	2.69



(a) AHL (b) ExF (c) MiF (d) MaF

Fig. 5. Critical diagrams of Holm's test at 95% confidence for the comparison between G3P-kEMLC and RA*k*EL. G3P-kEMLC-3 and G3P-kEMLC-V are indicated as G3P-k3 and G3P-kV, respectively.

the four metrics comparing G3P-kEMLC with RA*k*EL; in these diagrams, a line linking two methods indicates that their performance is statistically the same at 95% confidence.

Fig. 6 shows the average number of votes per label, $\overline{v}$, of the best configuration in each case. Note that, although RA*k*EL was executed with a large number of classifiers (such as n calculated for $\overline{v} = 10$ and for $\overline{v}$ equal to the best configuration of G3P-kEMLC), in most cases it performed better just with 6 votes on average for each label. We see that G3P-kEMLC is able to model and adjust the number of classifiers depending on each dataset, and is flexible to adapt to each specific case. In many cases, G3P-kEMLC obtained better results than RA*k*EL using less classifiers. Therefore, we also show that the number of classifiers used in the ensemble is not the only characteristic that contributes for our method to achieve good performance, but the selection of classifiers into an optimal tree-shaped ensemble structure is decisive for its performance.

B. G3P-kEMLC vs state-of-the-art

Once we have shown that the performance of G3P-kEMLC is significantly better than RA*k*EL, we next compare it with other state-of-the-art EMLCs. As in the previous experiment, Table III shows the average ranking values of each method on all datasets. In it, G3P-kEMLC-3 and G3P-kEMLC-V are indicated as G3P-3 and G3P-V respectively. We see that both G3P-kEMLC configurations always have better average ranking than the rest of EMLCs, except for SA, where EPS is better ranked.

Friedman's test determined that there existed significant differences in the performance of the algorithms for all metrics. Fig. 7 shows the critical diagrams of the Holm's test for all metrics. For AHL, ExF, and MiF, the performance of both G3P-kEMLC configurations is significantly better than all other methods. For MaF, the performance of G3P-kEMLC-V is statistically comparable to EBR, while for SA, RF-PCT is the only method that perform significantly worse than both EPS and G3P-kEMLC-V.

Therefore, we demonstrate that given the optimal structure of the ensemble that G3P-kEMLC obtains, independently of the configuration used, it outperformed state-of-the-art EMLCs.

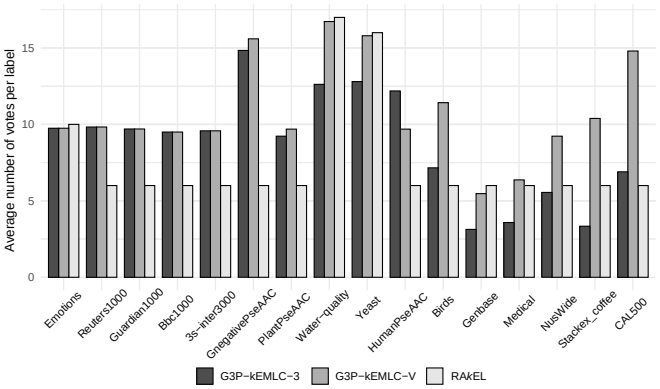
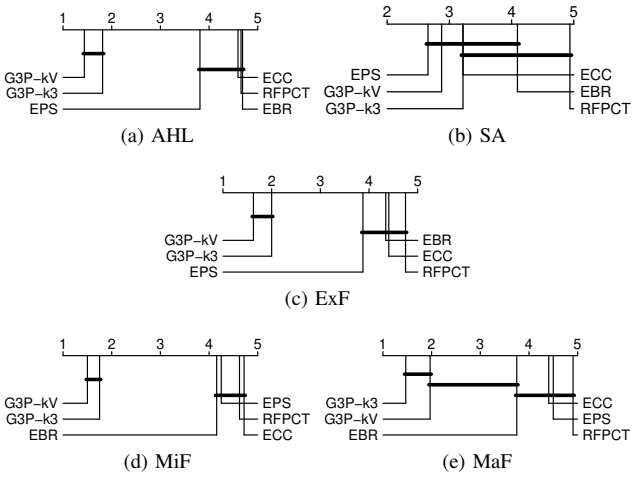


Fig. 6. Average number of votes per label in each dataset.

TABLE III
AVERAGE RANKINGS IN THE COMPARISON AMONG G3P-KEMLC AND STATE-OF-THE-ART EMLCS.

	G3P-3	G3P-V	ECC	EBR	EPS	RF-PCT
AHL	1.81	**1.44**	4.59	4.69	3.81	4.66
SA	3.22	2.88	3.22	4.09	**2.66**	4.94
ExF	2.00	**1.63**	4.41	4.34	3.88	4.75
MiF	1.75	**1.50**	4.72	4.16	4.25	4.63
MaF	**1.47**	1.97	4.41	3.75	4.50	4.91



(a) AHL (b) SA (c) ExF (d) MiF (e) MaF

Fig. 7. Critical diagrams of Holm's test at 95% confidence for the comparison between G3P-kEMLC and state-of-the-art EMLCs. G3P-kEMLC-3 and G3P-kEMLC-V are indicated as G3P-k3 and G3P-kV, respectively.

VI. Conclusions

In this paper, we introduced a method based on G3P for building EMLCs. Given a pool of multi-label classifiers, each focused on a subset of k labels, where k could be different for each, the algorithm evolves individuals representing a tree-shaped ensemble. At each node of the ensemble, predictions of children nodes are combined, while the leaves represent any multi-label classifier of the pool. The experimental study on 16 multi-label dataset using 5 evaluation metrics yielded promising results, demonstrating that the fact of building a tree-shaped ensemble where not all members have the same importance in the final prediction not only outperformed RAkEL (which also is focused on small subsets of the labels) but also the other state-of-the-art EMLCs, obtaining the model in acceptable time. Moreover, the proposed method is flexible and able to adapt the number of members of the ensemble according to each specific case.

In the future, we will explore other types of non-terminal nodes to combine the predictions, which also may use different types or number of child nodes. Further, we will develop a heuristic or evolutionary process to get a pool of multi-label classifiers that are accurate, diverse, and have an optimal size of the k-labelsets.

Acknowledgment

This research was supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund, project TIN2017-83445-P. This research was also supported by the Spanish Ministry of Education under FPU Grant FPU15/02948.

References

[1] E. Gibaja and S. Ventura, "Multi-label learning: a review of the state of the art and ongoing research," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 4, no. 6, pp. 411–444, 2014.

[2] H. Shao, G. Li, G. Liu, and Y. Wang, "Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine," *Science China Information Sciences*, vol. 56, no. 5, pp. 1–13, 2013.

[3] J. Xu, "Fast multi-label core vector machine," *Pattern Recognition*, vol. 46, no. 3, pp. 885–898, 2013.

[4] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective and efficient multilabel classification in domains with large number of labels," in *Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, 2008, pp. 53–59.

[5] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, no. 3, pp. 335–359, 2011.

[6] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multi-label classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079–1089, 2011.

[7] J. M. Moyano, E. L. Gibaja, K. J. Cios, and S. Ventura, "Review of ensembles of multi-label classifiers: Models, experimental study and prospects," *Information Fusion*, vol. 44, pp. 33 – 45, 2018.

[8] G. Tsoumakas, I. Partalas, and I. Vlahavas, "A taxonomy and short review of ensemble selection," in *Workshop on Supervised and Unsupervised Ensemble Methods and Their Applications*, 2008, pp. 1–6.

[9] Y. Bi, "The impact of diversity on the accuracy of evidential classifier ensembles," *International Journal of Approximate Reasoning*, vol. 53, no. 4, pp. 584 – 607, 2012.

[10] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 2, pp. 121–144, 2009.

[11] O. Gharroudi, H. Elghazel, and A. Aussem, "Ensemble Multi-label Classification: A Comparative Study on Threshold Selection and Voting Methods," in *IEEE International Conference on Tools with Artificial Intelligence,*, 2015, pp. 377–384.

[12] G. Tsoumakas, I. Katakis, and I. Vlahavas, *Data Mining and Knowledge Discovery Handbook, Part 6*. Springer, 2010, ch. Mining Multi-label Data, pp. 667–685.

[13] J. Read, B. Pfahringer, and G. Holmes, "Multi-label classification using ensembles of pruned sets," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 995–1000.

[14] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[15] D. Kocev, C. Vens, J. Struyf, and S. Džeroski, "Ensembles of multi-objective decision trees," in *European conference on machine learning*, 2007, pp. 624–631.

[16] H. Blockeel, L. D. Raedt, and J. Ramon, "Top-down induction of clustering trees," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 55–63.

[17] R. I. Mckay, N. X. Hoai, P. A. Whigham, Y. Shan, and M. O'neill, "Grammar-based genetic programming: a survey," *Genetic Programming and Evolvable Machines*, vol. 11, no. 3-4, pp. 365–396, 2010.

[18] A. Tsakonas, G. Dounias, M. Doumpos, and C. Zopounidis, "Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming," *Expert Systems with Applications*, vol. 30, no. 3, pp. 449–461, 2006.

[19] A. Zafra and S. Ventura, "Multi-instance genetic programming for predicting student performance in web based educational environments," *Applied Soft Computing*, vol. 12, no. 8, pp. 2693–2706, 2012.

[20] J. M. Luna, J. R. Romero, C. Romero, and S. Ventura, "On the use of genetic programming for mining comprehensible rules in subgroup discovery," *IEEE transactions on cybernetics*, vol. 44, no. 12, pp. 2329–2341, 2014.

[21] A. Tahmassebi and A. H. Gandomi, "Genetic programming based on error decomposition: A big data approach," in *Genetic Programming Theory and Practice XV*. Springer, 2018, pp. 135–147.

[22] A. Cano, A. Zafra, E. L. Gibaja, and S. Ventura, "A grammar-guided genetic programming algorithm for multi-label classification," in *European Conference on Genetic Programming*. Springer, 2013, pp. 217–228.

[23] A. G. C. de Sá, A. A. Freitas, and G. L. Pappa, "Automated selection and configuration of multi-label classification algorithms with grammar-based genetic programming," in *Parallel Problem Solving from Nature – PPSN XV*, 2018, pp. 308–320.

[24] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information sciences*, vol. 250, pp. 113–141, 2013.

[25] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Computing Surveys*, vol. 47, no. 3, 2015.

[26] J. M. Moyano, E. L. Gibaja, and S. Ventura, "MLDA: A tool for analyzing multi-label datasets," *Knowledge-Based Systems*, vol. 121, pp. 1–3, 2017.

[27] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "Mlenn: A first approach to heuristic multilabel undersampling," in *Intelligent Data Engineering and Automated Learning – IDEAL 2014*, 2014, pp. 1–9.

[28] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás, "JCLEC: a Java framework for evolutionary computation," *Soft Computing*, vol. 12, no. 4, pp. 381–392, 2008.

[29] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, "Mulan: A Java library for multi-label learning," *Journal of Machine Learning Research*, vol. 12, pp. 2411–2414, 2011.

[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[31] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

[32] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Ann. Math. Statist.*, vol. 11, no. 1, pp. 86–92, 03 1940.

[33] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.

References

[1] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[2] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1):1–39, 2010.

[3] Michal Wozniak, Manuel Graña, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3 – 17, 2014. Special Issue on Information Fusion in Hybrid Intelligent Fusion Systems.

[4] Lior Rokach. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics and Data Analysis*, 53(12):4046 – 4072, 2009.

[5] Joao Mendes-Moreira, Carlos Soares, Alípio Mário Jorge, and Jorge Freire De Sousa. Ensemble approaches for regression: A survey. *Acm computing surveys (csur)*, 45(1):1–40, 2012.

[6] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011.

[7] William Leigh, Russell Purvis, and James M. Ragusa. Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support. *Decision Support Systems*, 32(4):361 – 377, 2002.

[8] Aik Choon Tan, David Gilbert, and Yves Deville. Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics*, 14:206–217, 2003.

[9] Paul Mangiameli, David West, and Rohit Rampal. Model selection for medical diagnosis decision support systems. *Decision Support Systems*, 36(3):247 – 259, 2004.

[10] Hewi-Jen Lin, Yang-Ta Kao, Fu-Wen Yang, and Patrick S. P. Wang. Content-based image retrieval trained by adaboost for mobile application. *International Journal of Pattern Recognition and Artificial Intelligence*, 20(04):525–541, 2006.

[11] Thomas G. Dietterich. *Ensemble Methods in Machine Learning*, pages 1–15. Springer Berlin Heidelberg, 2000.

[12] L.I. Kuncheva, C.J. Whitaker, C.A. Shipp, and R.P.W. Duin. Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis & Applications*, 6(1):22–31, 2003.

[13] Yaxin Bi. The impact of diversity on the accuracy of evidential classifier ensembles. *International Journal of Approximate Reasoning*, 53(4):584 – 607, 2012.

[14] David W Opitz and Jude W Shavlik. Actively searching for an effective neural network ensemble. *Connection Science*, 8(3-4):337–354, 1996.

[15] David H. Wolpert. Stacked generalization. *Neural Networks*, (5):241–259, 1992.

[16] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, (7):179–188, 1936.

[17] K. Lang. The 20 newsgroups data set. http://qwone.com/~jason/20Newsgroups/. Last access: 02-05-2020.

[18] Forrest Briggs, Yonghong Huang, Raviv Raich, Konstantinos Eftaxias, Zhong Lei, William Cukierski, Sarah Frey Hadley, Adam Hadley, Matthew Betts, Xiaoli Z. Fern, Jed Irvine, Lawrence Neal, Anil Thomas, Gábor Fodor, Grigorios

Tsoumakas, Hong Wei Ng, Thi Ngoc Tho Nguyen, Heikki Huttunen, Pekka Ruusuvuori, Tapio Manninen, Aleksandr Diment, Tuomas Virtanen, Julien Marzat, Joseph Defretin, Dave Callender, Chris Hurlburt, Ken Larrey, and Maxim Milakov. The 9th annual MLSP competition: New methods for acoustic classification of multiple simultaneous bird species in a noisy environment. In *IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2013, Southampton, United Kingdom, September 22-25, 2013*, pages 1–8, 2013.

[19] Adriano Rivolli, Larissa C Parker, and Andre CPLF de Carvalho. Food truck recommendation using multi-label classification. In *EPIA Conference on Artificial Intelligence*, pages 585–596. Springer, 2017.

[20] John P. Pestian, Christopher Brew, Pawel Matykiewicz, D. J. Hovermale, Neil Johnson, K. Bretonnel Cohen, and Wodzislaw Duch. A shared task involving multi-label classification of clinical free text. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing (BioNLP '07)*, pages 97–104, 2007.

[21] André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 681–687, 2001.

[22] Eva Gibaja and Sebastián Ventura. Multi-label learning: a review of the state of the art and ongoing research. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(6):411–444, 2014.

[23] E Loza and J Fürnkranz. Efficient multilabel classification algorithms for large-scale problems in the legal domain. In *Semantic Processing of Legal Texts*, volume 6036, pages 192–215, 2010.

[24] F. Zou, Y. Liu, H. Wang, J. Song, J. Shao, K. Zhou, and S. Zheng. Multi-view multi-label learning for image annotation. *Multimedia Tools and Applications*, 2015.

[25] K Trohidis, G Tsoumakas, G Kalliris, and I Vlahavas. Multi-label classification of music into emotions. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008)*, pages 325–330, 2008.

[26] M Boutell, J Luo, X Shen, and C Brown. Learning multi-label scene classification. *Pattern recognition*, 37:1757–1771, 2004.

[27] Jingdong Wang, Yinghai Zhao, Xiuqing Wu, and Xian-Sheng Hua. A transductive multi-label learning approach for video concept detection. *Pattern Recognition*, 44(10–11):2274 – 2286, 2011.

[28] Z Barutcuoglu, RE Schapire, and OG Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22:830–836, 2006.

[29] Min-Ling Zhang and Zhi-Hua Zhou. Multi-label neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18:1338–1351, 2006.

[30] L Tang and H Liu. Scalable learning of collective behavior based on sparse social dimensions. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*, pages 1107–1116, 2009.

[31] Ouadie Gharroudi, Haytham Elghazel, and Alex Aussem. Ensemble Multi-label Classification: A Comparative Study on Threshold Selection and Voting Methods. In *IEEE International Conference on Tools with Artificial Intelligence,*, pages 377–384, 2015.

[32] Min-Ling Zhang and Kun Zhang. Multi-label learning by exploiting label dependency. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 999–1008, 2010.

[33] Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, 2012.

[34] G Tsoumakas, I Katakis, and I Vlahavas. *Data Mining and Knowledge Discovery Handbook, Part 6*, chapter Mining Multi-label Data, pages 667–685. Springer, 2010.

[35] M. Zhang and L. Wu. LIFT: Multi-label learning with label-specific features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):107–120, 2015.

[36] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):335–359, 2011.

[37] E.C. Goncalves, Alexandre Plastino, and Alex A. Freitas. A genetic algorithm for optimizing the label ordering in multi-label classifier chains. In *IEEE 25th International Conference on Tools with Artificial Intelligence*, pages 469–476. IEEE Computer Society Conference Publishing Services (CPS), 2013.

[38] J Read. A pruned problem transformation method for multi-label classification. In *Proceedings of the NZ Computer Science Research Student Conference*, pages 143–150, 2008.

[39] Lena Tenenboim-Chekina, Lior Rokach, and Bracha Shapira. Identification of label dependencies for multi-label classification. In *Working Notes of the Second International Workshop on Learning from Multi-Label Data*, pages 53–60, 2010.

[40] Priscilia E Greenwood and Michael S Nikulin. A guide to chi-squared testing. *Wiley-Interscience*, 280, 1996.

[41] Hendrik Blockeel, Luc De Raedt, and Jan Ramon. Top-down induction of clustering trees. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 55–63, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[42] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.

[43] Min-Ling Zhang and Zhi-Hua Zhou. A k-Nearest Neighbor Based Algorithm for Multi-label Classification. In *Proceedings of the IEEE International Conference on Granular Computing (GrC)*, volume 2, pages 718–721, Beijing, China, July 2005. The IEEE Computational Intelligence Society.

[44] Weiwei Cheng and Eyke Hullermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.

[45] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

[46] Nan Zhang, Shifei Ding, and Jian Zhang. Multi layer ELM-RBF for multi-label learning. *Applied Soft Computing*, 43:535 – 545, 2016.

[47] G Tsoumakas, A Dimou, E Spyromitros, V Mezaris, I Kompatsiaris, and I Vlahavas. Correlation-based pruning of stacked binary relevance models for multi-label learning. In *1st International Workshop on Learning from Multi-Label Data (MLD'09)*, pages 101–116, 2009.

[48] G Tsoumakas, I Katakis, and I Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, pages 53–59, 2008.

[49] Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39:135–168, 2000.

[50] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.

[51] Chen Lin, Wenqiang Chen, Cheng Qiu, Yunfeng Wu, Sridhar Krishnan, and Quan Zou. LibD3C: Ensemble classifiers with a clustering and dynamic selection strategy. *Neurocomputing*, 123:424 – 435, 2014.

[52] Jesse Read, Bernhard Pfahringer, and Geoff Holmes. Multi-label classification using ensembles of pruned sets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 995–1000. IEEE, 2008.

[53] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Random k-labelsets for multi-label classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011.

[54] Gulisong Nasierding, Abbas Z. Kouzani, and Grigorios Tsoumakas. A triple-random ensemble classification method for mining multi-label data. In *ICDMW 2010, The 10th IEEE International Conference on Data Mining Workshops, Sydney, Australia, 13 December 2010*, pages 49–56, 2010.

[55] Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Džeroski. Ensembles of multi-objective decision trees. In *European conference on machine learning*, pages 624–631, 2007.

[56] Lior Rokach. Decision forest: Twenty years of research. *Information Fusion*, 27:111 – 125, 2016.

[57] Gulisong Nasierding, Grigorios Tsoumakas, and Abbas Z. Kouzani. Clustering based multi-label classification for image annotation and retrieval. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11-14 October 2009*, pages 4514–4519, 2009.

[58] Bin Liu and Grigorios Tsoumakas. Synthetic oversampling of multi-label data based on local label distribution. *CoRR*, abs/1905.00609, 2019.

[59] Pascal Brandt, Deshendran Moodley, Anban W. Pillay, Christopher J. Seebregts, and Tulio de Oliveira. *An Investigation of Classification Algorithms for Predicting HIV Drug Resistance without Genotype Resistance Testing*, pages 236–253. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[60] G. Nasierding and A.Z. Kouzani. Empirical study of multi-label classification methods for image annotation and retrieval. pages 617–622, 2010.

[61] Jesse Read. Scalable multi-label classification. *PhD Thesis, University of Waikato*, 2010.

[62] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007.

[63] F. Charte, A. Rivera, M.J. Del Jesus, and F. Herrera. A first approach to deal with imbalance in multi-label datasets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8073 LNAI:150–160, 2013.

[64] Jacob Cohen, Patricia Cohen, Stephen G. West, and Leona S. Aiken. *Applied Multiple Regression / Correlation Analysis for the Behavioral Sciences*. 2002.

[65] L Chekina, L Rokach, and B Shapira. Meta-learning for selecting a multi-label classification algorithm. pages 220–227, 2011.

[66] Eva Gibaja and Sebastian Ventura. A tutorial on multilabel learning. *ACM Computing Surveys*, 47(3), 2015.

[67] Derek Greene and Pádraig Cunningham. A matrix factorization approach for integrating multiple data views. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part I*, ECML PKDD '09, pages 423–438, 2009.

[68] Douglas Turnbull, Luke Barrington, David Torres, and Gert Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):467–476, 2008.

[69] H. Shao, G.Z. Li, G.P. Liu, and Y.Q. Wang. Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine. *Science China Information Sciences*, 56(5):1–13, 2013.

[70] Jianhua Xu, Jiali Liu, Jing Yin, and Chengyu Sun. A multi-label feature extraction algorithm via maximizing feature variance and feature-label dependence simultaneously. *Knowledge-Based Systems*, 98:172 – 184, 2016.

[71] S. Diplaris, G. Tsoumakas, P. Mitkas, and I. Vlahavas. Protein classification with multiple algorithms. In *Proc. 10th Panhellenic Conference on Informatics (PCI 2005)*, pages 448–456, 2005.

[72] C.G.M. Snoek, M.Worring, J.C. van Gemert, J.-M. Geusebroek, and A.W.M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of ACM Multimedia*, pages 421–430, 2006.

[73] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*, page 48, 2009.

[74] Francisco Charte and David Charte. Working with multilabel datasets in R: The mldr package. *The R Journal*, 7(2):149–162, dec 2015.

[75] Ashok Srivastava and Brett Zane-Ulman. Discovering recurring anomalies in text reports regarding complex space systems. In *IEEE Aerospace Conference*, volume 37, 2005.

[76] H. Blockeel, S. Džeroski, and J. Grbović. Simultaneous prediction of multiple chemical parameters of river water quality with tilde. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1704:32–40, 1999.

[77] H. Sajnani, V. Saini, K. Kumar, E. Gabrielova, P. Choudary, and C. Lopes. Classifying yelp reviews into relevant categories. http://www.ics.uci.edu/~vpsaini/. Last access: 04-05-2020.

[78] Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1):5–45, 2012.

[79] Rafael B. Pereira, Alexandre Plastino, Bianca Zadrozny, and Luiz H.C. Merschmann. Correlation analysis of performance measures for multi-label classification. *Information Processing & Management*, 54(3):359 – 369, 2018.

[80] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. Mulan: A Java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.

[81] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[82] Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. MEKA: A multi-label/multi-target extension to Weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016.

[83] Piotr Szymański and Tomasz Kajdanowicz. A scikit-based python environment for performing multi-label classification. *arXiv preprint arXiv:1702.01460*, 2017.

[84] Sebastián Ventura and José María Luna. *Pattern mining with evolutionary algorithms*. Springer, 2016.

[85] Satyasai Jagannath Nanda and Ganapati Panda. A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm and Evolutionary Computation*, 16:1 – 18, 2014.

[86] Mohammad Taradeh, Majdi Mafarja, Ali Asghar Heidari, Hossam Faris, Ibrahim Aljarah, Seyedali Mirjalili, and Hamido Fujita. An evolutionary gravitational search-based feature selection. *Information Sciences*, 497:219 – 239, 2019.

[87] Michael Affenzeller, Stefan Wagner, Stephan Winkler, and Andreas Beham. *Genetic algorithms and genetic programming: modern concepts and practical applications*. Crc Press, 2009.

[88] Agoston E Eiben and James E Smith. *Introduction to evolutionary computing*. Springer, 2003.

[89] DE Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[90] Riccardo Poli, William B Langdon, Nicholas F McPhee, and John R Koza. *A field guide to genetic programming*. Lulu. com, 2008.

[91] Cândida Ferreira. *Gene expression programming: mathematical modeling by an artificial intelligence*, volume 21. Springer, 2006.

[92] Mitchell A Potter and Kenneth A De Jong. A cooperative coevolutionary approach to function optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 249–257, 1994.

[93] Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm intelligence*, 1(1):33–57, 2007.

[94] Sebastián Ventura, Cristóbal Romero, Amelia Zafra, José A. Delgado, and César Hervás. JCLEC: a Java framework for evolutionary computation. *Soft Computing*, 12(4):381–392, 2008.

[95] Noureddine-Yassine Nair-Benrekia, Pascale Kuntz, and Frank Meyer. Learning from multi-label data with interactivity constraints: An extensive experimental study. *Expert Systems with Applications*, 42(13):5723 – 5736, 2015.

[96] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.

Vita

Jose María Moyano Murillo was born on October 18th, 1992, in El Viso, Córdoba, Spain. He received his B.Sc. and M.Sc. in Computer Science from the University of Córdoba, Spain, in 2014 and 2016 respectively. He has developed his thesis enrolled in the dual Ph.D. program at both University of Córdoba and Virginia Commonwealth University, granted with a FPU Grant given by the Spanish Ministry of Education. His research is mainly focused on ensemble learning algorithms for multi-label classification.

Journal papers:

[J1] Eva L. Gibaja, **Jose M. Moyano**, and Sebastián Ventura. (2016). An ensemble-based approach for multi-view multi-label classification. *Progress in Artificial Intelligence*, 5(4): 251-259. DOI: 10.1007/s13748-016-0098-9.

[J2] **Jose M. Moyano**, Eva L. Gibaja, and Sebastián Ventura. (2017). MLDA: A tool for analyzing multi-label datasets. *Knowledge-Based Systems*, 121: 1-3. DOI: 10.1016/j.knosys.2017.01.018.

[J3] Isaac Triguero, Sergio González, **Jose M. Moyano**, Salvador García, Jesús Alcalá-Fdez, Julián Luengo, Alberto Fernández, Maria José del Jesus, Luciano Sánchez, and Francisco Herrera. (2017). KEEL 3.0: An open source software for multi-stage analysis in data mining. *International Journal of Computational Intelligence Systems*, 10(1): 1238-1249. DOI: 10.2991/ijcis.10.1.82.

[J4] **Jose M. Moyano**, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura. (2018). Review of ensembles of multi-label classifiers: Models, experimental study and prospects. *Information Fusion*, 44: 33-45. DOI: 10.1016/j.inffus.2017.12.001.

[J5] **Jose M. Moyano**, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura. (2019). An Evolutionary approach to build ensembles of multi-label classifiers. *Information Fusion*, 50: 168-180. DOI: 10.1016/j.inffus.2018.11.013.

[J6] **Jose M. Moyano**, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura. (2020). Combining multi-label classifiers based on projections of the output space using Evolutionary algorithms. *Knowledge-Based Systems*, 196:105770. DOI: 10.1016/j.knosys.2020.105770.

Conference papers:

[C1] **Jose M. Moyano**, Eva L. Gibaja, Alberto Cano, Jose M. Luna, and Sebastián Ventura. (2015). Algoritmo evolutivo para optimizar ensembles de clasificadores multi-etiqueta. In *X Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB 2015*, 219-225.

[C2] Eva L. Gibaja, **Jose M. Moyano**, and Sebastián Ventura. (2015). Combinación de vistas para clasificación multi-etiqueta: estudio preliminar. In *VII Simposio de Teoría y Aplicaciones de la Minería de Datos, TAMIDA 2015, CAEPIA 2015*, 759-768.

[C3] **Jose M. Moyano**, Eva L. Gibaja, and Sebastián Ventura. (2015). Diseño automático de multi-clasificadores basados en proyecciones de etiquetas. In *XVI Conferencia de la Asociación Española para la Inteligencia Artificial, CAEPIA 2015*, 355-365.

[C4] **Jose M. Moyano** and Luciano Sánchez. (2016). RKEEL: Using KEEL in R code. In *2016 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2016*, 257-264. DOI: 10.1109/FUZZ-IEEE.2016.7737695.

[C5] **Jose M. Moyano**, Eva L. Gibaja, and Sebastián Ventura. (2016). Una herramienta para analizar conjuntos de datos multi-etiqueta. In *XVII Conferencia de la Asociación Española para la Inteligencia Artificial, CAEPIA 2016*, 857-866.

[C6] Oliver Sánchez, **Jose M. Moyano**, Luciano Sánchez, and Jesús Alcalá-Fdez. (2017). Mining association rules in R using the package RKEEL. In *2017 IEEE*

International Conference on Fuzzy Systems, FUZZ-IEEE 2017, 1-6. DOI: 10.1109/FUZZ-IEEE.2017.8015572.

[C7] **Jose M. Moyano**, Eva L. Gibaja, and Sebastián Ventura. (2017). An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains. In *2017 IEEE Congress on Evolutionary Computation, CEC 2017*, 2015-2021. DOI: 10.1109/CEC.2017.7969548.

[C8] Óscar Reyes, **Jose M. Moyano**, Jose M. Luna, and Sebastián Ventura. (2018). A gene expression programming method for multi-target regression. In *International Conference on Learning and Optimization Algorithms: Theory and Applications, LOPAL 2018*, 2:1-2:6. DOI: 10.1145/3230905.3230910.

[C9] Óscar Reyes, Jose M. Luna, **Jose M. Moyano**, Eduardo Pérez, and Sebastián Ventura. (2018). Resolución de problemas biomédicos mediante técnicas de extracción de conocimiento. In *XVIII Conferencia de la Asociación Española para la Inteligencia Artificial, CAEPIA 2018*, 1252-1257.

[C10] Óscar Reyes, **Jose M. Moyano**, Jose M. Luna, and Sebastián Ventura. (2018). Resolviendo el problema de regresión multi-salida mediante Gene Expression Programming. In *IX Simposio de Teoría y Aplicaciones de la Minería de Datos, TAMIDA 2018, CAEPIA 2018*, 907-912.

[C11] Óscar Reyes, **Jose M. Moyano**, Antonio Rivero-Juárez, Raúl M. Luque, Antonio Rivero, Justo Castaño, and Sebastián Ventura. (2018). Extracción de factores relevantes en el análisis de datos biomédicos: una metodología basada en técnicas de aprendizaje supervisado. In *IX Simposio de Teoría y Aplicaciones de la Minería de Datos, TAMIDA 2018, CAEPIA 2018*, 861-866.

[C12] **Jose M. Moyano**, Eva L. Gibaja, Sebastián Ventura, and Alberto Cano. (2019). Speeding up classifier chains in multi-label classification. In *4th International Conference on Internet of Things, Big Data and Security, IoTBDS 2019*, 29-37. DOI: 10.5220/0007614200290037.

[C13] **Jose M. Moyano**, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura. (2020). Generating ensembles of multi-label classifiers using cooperative coevolution-

ary algorithms. In *24th European Conference on Artificial Intelligence, ECAI 2020*. Accepted paper.

[C14] **Jose M. Moyano**, Eva L. Gibaja, Krzysztof J. Cios, and Sebastián Ventura. (2020). Tree-shaped ensemble of multi-label classifiers using grammar-guided genetic programming. In *2020 IEEE Congress on Evolutionary Computation, CEC 2020*. Accepted paper.