24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

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 (RA*k*EL) [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 RA*k*EL 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 RA*k*EL. However, unlike RA*k*EL, 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 RA*k*EL 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.

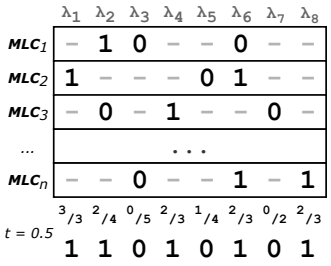


Figure 1: Example of the prediction phase of the EMLC.

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.

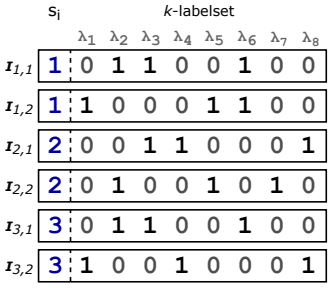


Figure 2: Example of individuals of the CCEA. Each individual includes both the index of the subpopulation as well as the labels present in its *k*-labelset.

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

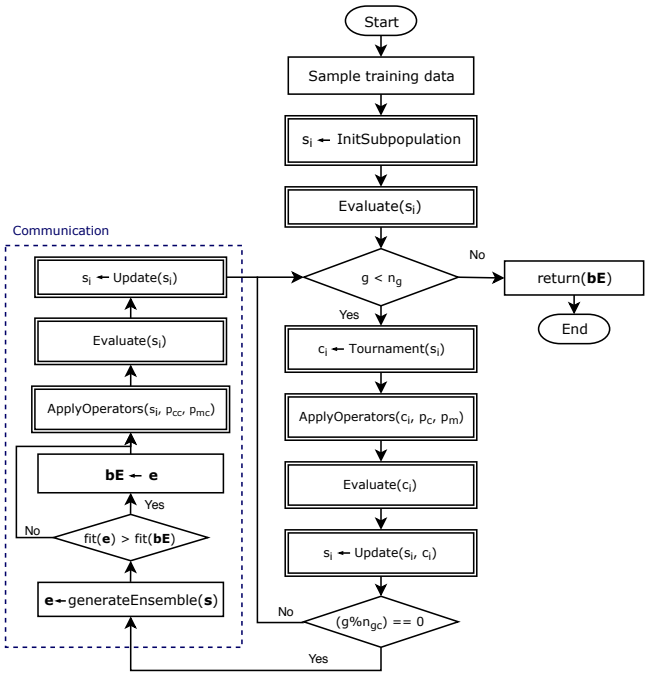


Figure 3: Main steps of the CCEA. Boxes with double lines indicate that the process is performed independently for each subpopulation. The region within the dashed line indicates the communication between subpopulations.

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.

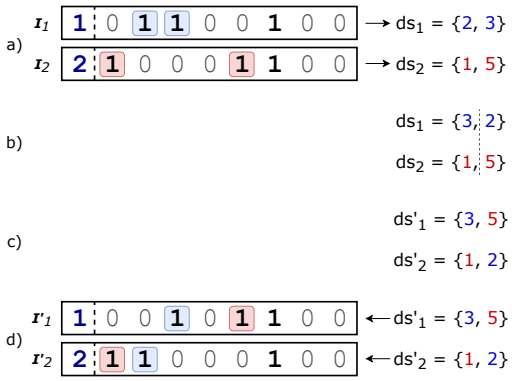


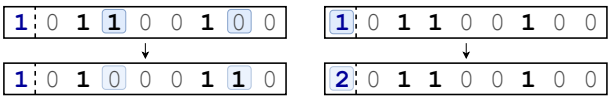
Figure 4: Example of crossover operator.

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].

[7] Additional material available at http://www.uco.es/kdis/CCEA

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.

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**

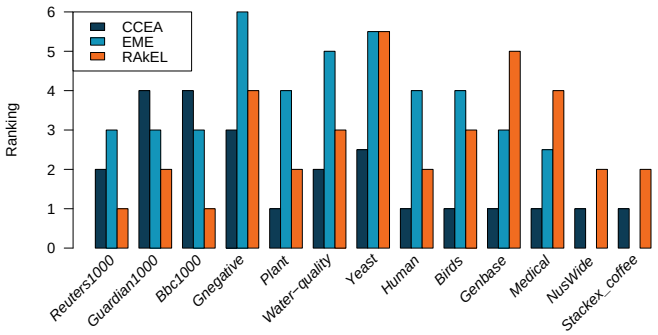
Table 4: Results of Bonferroni-Dunn's test.

	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**

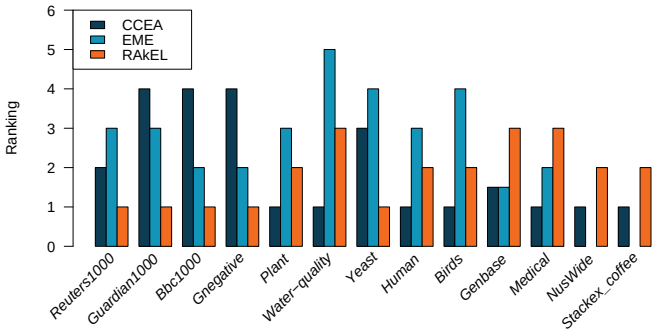
From the results we can reach several conclusions. First, the proposed CCEA is able to outperform EME. The fact of evolving individuals as separate members of the ensemble instead of using the entire ensemble allows the CCEA to build an ensemble with more promising members, while considering both performance 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).