Simpler is Better: a Novel Genetic Algorithm to Induce Compact Multi-label Chain Classifiers

Eduardo C. Gonçalves
Institute of Computing
Universidade Federal Fluminense
Niterói, Brazil
egoncalves@ic.uff.br

Alexandre Plastino
Institute of Computing
Universidade Federal Fluminense
Niterói, Brazil
plastino@ic.uff.br

Alex A. Freitas
School of Computing
University of Kent at Canterbury
Kent, CT2 7NF, UK
A.A.Freitas@ukc.ac.uk

ABSTRACT

Multi-label classification (MLC) is the task of assigning multiple class labels to an object based on the features that describe the object. One of the most effective MLC methods is known as Classifier Chains (CC). This approach consists in training q binary classifiers linked in a chain, $\{y_1 \rightarrow y_2 \rightarrow ... \rightarrow y_q\}$, with each responsible for classifying a specific label in $\{l_1, l_2, ..., l_q\}$. The chaining mechanism allows each individual classifier to incorporate the predictions of the previous ones as additional information at classification time. Thus, possible correlations among labels can be automatically exploited. Nevertheless, CC suffers from two important drawbacks: (i) the label ordering is decided at random, although it usually has a strong effect on predictive accuracy; (ii) all labels are inserted into the chain, although some of them might carry irrelevant information to discriminate the others. In this paper we tackle both problems at once, by proposing a novel genetic algorithm capable of searching for a single optimized label ordering, while at the same time taking into consideration the utilization of partial chains. Experiments on benchmark datasets demonstrate that our approach is able to produce models that are both simpler and more accurate.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

Keywords

multi-label classification, genetic algorithms, classifier chains

1. INTRODUCTION

In recent years, multi-label classification (MLC) [25, 29] has become one of the most active research topics in the fields of data mining and machine learning. In this problem, each object of a dataset may belong to multiple class labels and the goal is to learn a system that can infer the correct labels of new, previously unseen, objects. A popular application is the classification of free text documents. For instance, an MLC system trained to infer movie genres according to their plots, could process a document containing the text summary of the Pedro Almodóvar's movie "Volver" and determine its genres as "Comedy", "Crime", and "Drama". Besides text classification, other modern applications of MLC include functional genomics (determining the multiple biological functions of genes), medical diagnosis (predicting the set of diseases a patient may develop) and direct marketing (recommendation of products for customers).

An important issue in MLC relates to the existence of dependence relationships among labels. For example, in the movie classification domain, intuitively a film is unlikely to be simultaneously considered as "Crime" and "Musical", since these two genres have a negative correlation. Analogously, the likelihood of a movie being labeled as "Crime" becomes stronger if it has been labeled as "Action" and "Thriller". Thus, it is expected that MLC methods capable of exploiting label dependencies should be more accurate. Indeed, a large number of recent proposals have concentrated efforts to tackle this issue by making use of diverse probabilistic techniques [13, 23, 24, 28].

In a different vein, the Classifier Chains (CC) method [22] employs a simpler non-probabilistic strategy to incorporate label dependencies into the classification process. In this approach, q binary classifiers are inserted at random order into a chain $\{y_1 \rightarrow y_2 \rightarrow ... \rightarrow y_q\}$, where each one is responsible for predicting a specific label in $\{l_1, l_2, ..., l_q\}$. The chain structure allows each binary classifier y_j to incorporate the labels inferred by the previous $y_1, ..., y_{j-1}$ classifiers as additional predictive information. Although the adopted principle is quite simple, a comprehensive empirical study [20] demonstrated that CC is able to outperform most of the other existing state-of-the-art MLC methods.

Nevertheless, the CC method suffers from two major drawbacks. First, it decides the label sequence at random. As demonstrated in [3], the use of distinct chain orderings can lead to large differences in the predictive accuracy of the model. Second, CC forces all labels to be present in the chain, despite the fact that some of them might carry redundant and irrelevant information. This might confuse the classification model instead of helping in discriminating the various other labels. To overcome these disadvantages, in this work we propose a novel Genetic Algorithm (GA) that performs a global search for an optimized chain (i.e., a label sequence that leads to an improvement on the predictive ac-

GECCO '15, July 11 - 15, 2015, Madrid, Spain

 ISBN 978-1-4503-3472-3/15/07...$15.00

DOI: http://dx.doi.org/10.1145/2739480.2754650

curacy of the CC model), by exploring partial chains with only a subset of labels. To the best of our knowledge, this is the first approach proposed with the explicit goal of finding an optimized partially chained model for MLC.

This paper is organized as follows. Section 2 gives a short overview on MLC. Section 3 reviews the original CC framework. Section 4 reviews work related to our proposal. Section 5 formally defines the concept of partially chained (PartCC) model for MLC. Section 6 describes the characteristics of our proposed GA. Section 7 reports the experimental results. Concluding remarks and future research directions are given in Section 8.

2. MULTI-LABEL CLASSIFICATION

The MLC problem can be defined as follows.

DEFINITION 1. *(Multi-label Classification Problem). Let $X = \{A_1, ..., A_d\}$ be a finite set of d predictive (input) attributes. Let $L = \{l_1, ..., l_q\}$ be a finite set of q possible class labels, where $q \geq 2$. Consider a training dataset $\mathcal{D}$ composed by N instances of the form $\{(x_1, Y_1), (x_2, Y_2), ..., (x_N, Y_N)\}$, where each x_i is a vector $\{(x_1, ..., x_d)\}$ that stores values for the d predictive attributes in X and each $Y_i \subseteq L$ is a subset of labels. The goal of the multi-label classification task is to learn a classifier (classification model) $h(x) \rightarrow Y$ from $\mathcal{D}$ that, given an unlabeled instance $t = (x, ?)$, predicts its set of labels (labelset) Y.*

The evaluation of MLC methods typically uses multiple performance measures, mainly because in MLC a result can be fully correct, fully wrong or partially correct. Hence, different performance measures provide the user with alternative analyses of the results, giving a better understanding about the actual predictive performance of a MLC method (an illustrative example is given in [11]).

In what follows, we introduce the measures used to assess the quality of the methods evaluated in our experiments. In the definitions, we use the following notation: n is the number of test instances; q is the number of labels; Y_i and Z_i represents, respectively, the actual and the predicted labelset of the i^{th} test instance.

The Exact Match (EM), defined in Eq. 1, assesses the percentage of instances fully correctly predicted in the test set. Consider that $I(true) = 1$ and $I(false) = 0$.

$$EM = \frac{1}{n} \sum_{i=1}^{n} I(Y_i = Z_i) \tag{1}$$

Note that EM can deal with results that are both fully wrong and fully correct. However, it is unable to evaluate partially correct results. To cope with this issue, we also use the Accuracy (ACC), F-Measure (FM) and Hamming Loss (HL) measures, respectively defined in Eqs. 2, 3, 4. In Eq. 4, the expression $Y_i \triangle Z_i$ gives the symmetric difference between Y_i and Z_i. For EM, ACC and FM, greater values indicate better performance; whilst for HL, smaller values indicate better performance.

$$ACC = \frac{1}{n} \sum_{i=1}^{n} \frac{|Z_i \cap Y_i|}{|Z_i \cup Y_i|} \tag{2}$$

$$FM = \frac{1}{n} \sum_{i=1}^{n} \frac{2 \times |Z_i \cap Y_i|}{|Z_i| + |Y_i|} \tag{3}$$

$$HL = \frac{1}{n} \sum_{i=1}^{n} \frac{|Z_i \triangle Y_i|}{q} \tag{4}$$

3. CLASSIFIER CHAINS

Among several MLC methods proposed in the literature [20, 25, 29], the Binary Relevance (BR) is perhaps the simplest one. In this approach, the MLC problem is transformed into q independent binary classification problems. One binary classifier is separately trained to predict each label. To infer the labelset of a new object, the BR model simply aggregates the labels positively predicted by each of the independent classifiers. Figure 1 shows a BR model, viewed as a graph, in a problem with four labels. The node X represents the set of predictive attributes whereas the nodes l_1, l_2, l_3, l_4 represent the target labels. The directed edges from X to each of the four labels indicate that, in the BR approach, the binary classifiers responsible for the prediction of each label are trained using exclusively the attributes in X as the set of input attributes. Hence, all binary classifiers in the group are isolated from each other, being unable to exchange information at both the training and the classification steps.

The BR strategy has several advantages. First it is simple. Second, it is algorithm independent, i.e., it enables abstraction from the underlying base algorithm for binary classification. This is a major advantage, because different classifiers (such as decision trees, SVM, Bayesian methods, etc. [27]) are more or less effective in different application domains. Finally, BR scales linearly with q and can be easily parallelized. Nonetheless, an obvious and critical drawback lies in that the BR model completely ignores the possible correlations among labels, as the binary classifiers make decisions independently from each other. Thus, the method is more suitable for problems where only a small number of labels exhibit correlation with each other.

Proposed in [22], the Classifier Chains (CC) method can be seen as a direct extension of the BR approach, capable of exploiting label dependencies though. Similar to BR, the CC method involves training a group of q binary classifiers. Nevertheless, instead of being kept isolated from each other, these q classifiers are linked in a chain structure, which allows each one to pass their predictions to the other binary classifiers ahead in the chain. In order to better explain the CC method, consider the graph depicted in Figure 2, representing a CC model with four labels chained in the sequence $\{l_1 \rightarrow l_2 \rightarrow l_3 \rightarrow l_4\}$. The directed edges in the graph indicate that the binary classifier y_1, which is responsible for classifying the first label in the chain (l_1), will be trained using solely the attributes that compose the attribute set X as its input attributes. However, the binary classifier y_2, re-

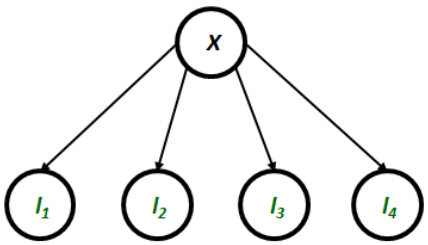

Figure 1: BR model

sponsible for the prediction of the second label in the chain (l_2), will be trained using X augmented with l_1 as the set of input attributes. Analogously, each subsequent classifier y_j will be trained using X augmented with the labels associated to the previous $j-1$ classifiers in the chain. Once the model is trained, the classification mechanism must be performed according to the chain sequence. To predict the labelset of a new object, q binary classifications are needed, with the process beginning at the first classifier in the sequence (y_1 in the example of Figure 2) and going along the chain. In this procedure, the classifier y_j predicts the relevance of label l_j, given the feature space augmented by the predictions carried out by the previous $j-1$ classifiers.

Although CC employs a direct and non-probabilistic approach to incorporate label dependencies into the classification model, it is considered one of the most effective MLC methods, in the sense that it has proven to be superior to state-of-the-art techniques in terms of predictive performance [20]. Besides this, CC still maintains most of the attractive characteristics of BR: it is algorithm independent, scales linearly with q and its training step can be easily parallelizable. Not surprisingly, CC has become one of the most adopted and representative frameworks for MLC [29].

4. RELATED WORK

A number of variations of the CC method have recently been proposed [3, 11, 16, 21, 24]. A common characteristic of these proposals is that they try to eliminate a key drawback in the original method: the fact that the label ordering is decided at random. In a recent study [3], an exhaustive experiment that compared the predictive performance of CC models built with thousands of distinct label orderings revealed that the differences in accuracy tend to be very large. The same study also revealed that, for many datasets, the original CC method (trained with a random order) might present a performance inferior to the BR method even in problems where most labels are correlated with each other. It is thus important to invest in algorithmic solutions to find an optimized chain order. Nonetheless, this is a difficult problem because of the enormous search space of $q!$ different possible label permutations.

Current extensions to the CC method make use of different approaches to solve the label sequence optimization problem (LSOP). The authors of the original CC model proposed the combination of random orders via an ensemble of classifier chains (ECC) in order to mitigate the effect of poorly ordered chains [22]. Differently, the approaches proposed in [3, 11, 16, 21] employ heuristic methods. In [16], the

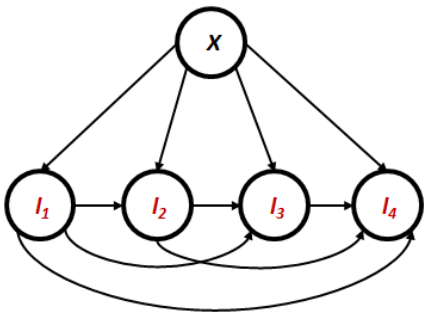

Figure 2: CC model

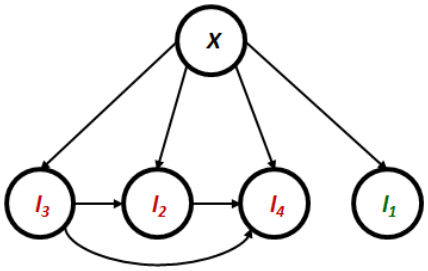

Figure 3: PartCC model

LSOP is addressed by performing a beam search over a tree in which every distinct path represents a different label permutation. The M2CC algorithm, described in [21], employs a double-Monte Carlo optimization technique to efficiently generate and evaluate a small population of distinct label sequences. The GACC method proposed in [11] performs a genetic algorithm search to solve the LSOP. In this technique, each chromosome represents a different label permutation. Crossover works by transferring sub-chains of random length between two individuals whilst mutation swaps pairs of labels of an individual. Finally, [3] proposes a lazy approach capable of searching for a distinct and more effective label sequence to each new instance t. In this strategy, for each instance in the training dataset, an effective label sequence is previously identified. At classification time, the label sequence for the new instance t is chosen based on the sequences associated with the instances in the training dataset that are more similar to t.

5. PARTIALLY CHAINED MODELS

The variations of the CC method aim at improving the model's effectiveness by handling the LSOP problem using different techniques. However, there is another important drawback in the original CC method that has been neglected in the literature. It corresponds to the fact that CC forces all labels to be present in the chain. None of the extensions have yet explored the idea of generating models defined by partial chains. In this context, the aim is to build a CC model in which one or more labels may be absent from the chain because their presence would lead to a decrease in the predictive accuracy. This is because some classifiers may pass redundant and irrelevant information, or wrongly predicted labels, along the chain, which might confuse the subsequent classifiers in the chain. Therefore, it might be interesting to remove these irrelevant or redundant labels from the chain structure (using independent binary classifiers for predicting each of them) and to create a partial chain with an optimized sequence using only the remaining labels.

An example of partially chained model (PartCC) for a MLC problem with four labels is shown in Figure 3. In this example, the classifiers associated to labels l_2, l_3 and l_4 are linked together in an optimized classifier chain model in the sequence $\{y_3 \rightarrow y_2 \rightarrow y_4\}$. The label l_1 is not in the chain. The intuition behind this representation is to indicate that l_1 will be treated by an independent binary classifier, because its presence in the chain would decrease the predictive accuracy of the multi-label classifier as a whole.

The PartCC model can be seen as an hybrid strategy between a BR model and a CC model using an optimized label sequence. Thus, it is expected to be effective in both

situations: when the majority of the labels are dependent on each other and also in the converse case, when most labels are independent. More interestingly, since it deals with partial chains, it is capable of producing simpler, more compact multi-label chain classifiers, offering gains in terms of efficiency. Furthermore, a PartCC model is particularly suitable for problems that require the construction of comprehensible predictive models [9], as the shorter sequence is more realistic for the representation of true label dependencies in comparison with a full length sequence. Nonetheless, the problem of finding an optimized PartCC model is much more challenging than the traditional LSOP as the search space is composed by an ordinary BR model (where all labels are "disconnected") plus the models formed by all possible chain permutations of length 2 to q. Consequently, while the search space in the LSOP has size equal to $q!$, the size of the search space in the problem of optimizing PartCC models is much higher, being given by Eq. 5.

$$1 + \sum_{r=2}^{q} \frac{q!}{(q-r)!} \tag{5}$$

In this paper, a genetic algorithm for learning optimized PartCC models is proposed: GA for Optimizing Partially-Chained Models (GA-PartCC). To the best of our knowledge, this is the first method proposed with the following two explicit goals: (i) defining the most relevant labels to be chained and (ii) searching for an optimized sequence to place them in regard to the improvement of the classification accuracy. Our main motivations for presenting a solution based on the GA paradigm are as follows:

- GAs have been widely employed to solve a large number of classification problems in the most distinct contexts and application domains [1, 2, 5, 6, 8, 18].
- GAs are a global search method capable of effectively exploring the extremely large search spaces associated to the partially chained optimization problem. Due to this, GAs tend to cope better with attribute interactions than greedy methods [6, 8].
- GAs have been successfully applied to solve optimization problems where a candidate solution is represented as a permutation, like the classical traveling salesman problem (TSP) [17] and the vehicle routing problem (VRP) [19]. Note, however, that LSOP, as a classification problem, involve prediction and overfitting issues, unlike the TSP and VRP problems.
- As the CC and BR approaches, GAs are also easily parallelizable.

6. THE GA-PARTCC ALGORITHM

6.1 Representation and Initial Population

In GA-PartCC, individuals are represented by variable-length lists. Each candidate solution specifies both a subset of labels and the order they are placed into the chain. For instance, the model in Figure 3 is encoded as the list $[l_3, l_2, l_4]$. Similarly, the models in Figures 1 and 2 are respectively encoded as [ ] (empty list) and $[l_1, l_2, l_3, l_4]$.

We adopted a simplified controlled approach for generating the initial population. Let P be the population size. The first individual is an empty chain (i.e., a BR model) and the second individual constitutes a complete CC model with the default chain sequence (as the one shown in Figure 2). The remainder individuals are generated as follows. First, individual lengths are randomly drawn following a uniform distribution in the range [2, q]. As a consequence, $P-2$ chromosomes with distinct lengths will be generated. For each of these chromosomes, starting from an empty chromosome and from the leftmost allele, the chain sequence is defined by randomly selecting integer numbers (representing the label indexes) according to a uniform distribution in the range [1, q]. Repeated integers are not allowed in a chromosome.

6.2 Lexicographic Fitness Function

We adopted a multi-objective lexicographic approach [7] to determine the fitness of the candidate solutions. In this technique, two or more objectives with distinct predetermined priorities are taken into consideration to define the quality of each chromosome. Consider the following example. Let c_i and c_j be two candidate solutions. In the lexicographic approach used in GA-PartCC, when comparing two chromosomes, the GA first tries to determine which one is better considering the highest priority objective. If c_i is not better than c_j, and vice-versa, then both are compared considering the second objective. The process is repeated until either a winner is found or all the criteria have been tested (in the later case, if no winner was found, one of the chromosomes is randomly chosen as winner).

In GA-PartCC, we only consider two objectives: predictive accuracy (first priority) and model simplicity (second priority). To assess the predictive accuracy of a chromosome, we use the Quality (fitness) function defined in [3, 11]. This function simultaneously takes into account the measures of Exact Match, Accuracy and Hamming Loss, respectively defined in Equations 1, 2 and 4. The Quality of a chromosome c_i is computed as:

$$Quality(c_i) = \frac{(1 - HL) + ACC + EM}{3} \tag{6}$$

If two chromosomes have the same value for the Quality function, they must be compared with regard to model's simplicity (second objective). We consider a solution c_i simpler than c_j if c_i encodes a chain sequence shorter than the one encoded in c_j. The rationale lies primarily in the Occam's Razor principle [4] which states that "given two models with the same generalization error, the simplest one should be preferred because simplicity is desirable in itself". Indeed, as discussed in Section 5, a shorter model offers two other important advantages. First, it is more efficient, as it involves a smaller number of attributes. Second, it gives higher fidelity for representing label dependencies, since only the most relevant labels with regard to the classification problem are present in the chain.

By comparison with the simple use of a weighted formula to combine the two objective values, the lexicographic approach has the advantage of avoiding the specification of ad-hoc numeric weights; it requires only the specification of the relative priority of the objectives, which is well-defined in the context of classification (accuracy clearly has priority over the chain size). By comparison with the Pareto dominance-based approach for multi-objective optimization, the lexicographic approach has the advantages of being simpler and avoiding the issue of how to choose one single solution to be

used in practice (out of all non-dominated solutions). Also, the usual Pareto approach would not allow us to specify that maximizing accuracy is more important than reducing the chain size, an important application-specific piece of knowledge that is naturally specified using the lexicographic approach.

The GA-PartCC follows the wrapper approach [6] in which the quality of an individual (candidate PartCC model) is determined by using the target MLC algorithm (i.e., the CC algorithm). The fitness function is calculated using only the training set, according to a holdout method that works as follows. First, the training set is partitioned into two mutually-exclusive subsets: building (2/3 split) and validation (1/3 split). Next, for each chromosome we build a PartCC model using only the building set, and then that model is evaluated with the validation set.

6.3 Selection Method

GA-PartCC uses the tournament selection method [6]. In this method, first k individuals are randomly drawn from the population, where k is a user-specified parameter called tournament size. Their fitness values are then compared according to the lexicographic approach presented in the previous subsection. The winner is the individual with the best lexicographic evaluation among the k participants. The selection procedure also implements elitism, in which a percentage of elite individuals are preserved from the previous generation according to their fitness values.

6.4 Genetic Operators

GA-PartCC implements crossover and mutation to deal with the two levels of representation encoded in the chromosomes (chain sequence and chain length).

The crossover operation consists in a modified version of the order crossover [5] method, which is often used in in GAs for permutation problems. The adaptation was necessary because the original method can deal only with chromosomes of the same length. In order to facilitate the explanation, consider the example shown in Figure 4. As the original method, our version of order crossover generates two children (represented by O_1 and O_2) from two parents (represented by P_1 and P_2). As shown in the figure, child O_1 must have the same length as P_2 and O_2 the same length as P_1. The crossover operates as follows. Two crossover points (represented by the two vertical thick lines in Figure 4) are chosen at random. The first step to generate O_1 is to copy the segment between the crossover points from P_1 into O_1 (Figure 4a). The second step consists in filling the remainder empty alleles in O_1 with genetic material from P_2 (Figure 4b). The procedure works as follows. Starting from the position next to the second crossover point (the fourth position in our example), the values that are present in P_2 but are not contained in O_1 are transferred to the empty alleles in O_1, wrapping around when the last position of both chromosomes is reached. O_2 is analogously generated.

The main advantages of the order crossover method used by GA-PartCC are the facts that it preserves the relative order in the parents in their children and always generates valid solutions. However, it does not create children with lengths difference form their parents' lengths, since the first child has the same length of the second parent and vice-versa. In order to increase the population's variability of length, in GA-PartCC, children resulting from crossover can

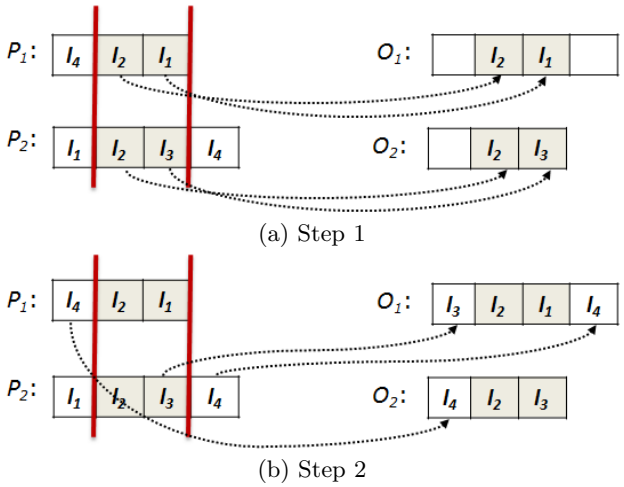

Figure 4: Adapted order crossover operation

be also subject to mutation, which consists in either inserting or removing a subchain. In the insertion procedure, a single insertion point is chosen at random in the current child. Next, a subchain with maximum length of 5% of q is randomly generated and inserted at that insertion point. The inserted sub-chain contains only labels that do not occur in the current child. The deletion procedure removes a segment between two randomly-chosen points.

7. EXPERIMENTS

7.1 Experimental Methodology

The performance of GA-PartCC was compared to the baseline methods BR and CC, and also against the GACC method [11], which can only deal with the optimization of full-length chains. We evaluated all methods using MULAN [26], an open-source package of Java classes for MLC that works on the top of the popular WEKA data mining tool [14]. The WEKA's J48 decision tree algorithm with default parameters was used as the base binary classification algorithm for all evaluated methods.

Table 1 presents the nine distinct benchmark datasets used in our experiments. In this table, the second column (q) presents the number of labels and the third (N) the number of instances. Observe that the datasets vary considerably in the number of labels, number of instances and application domain. Further information about the datasets can be found at the MULAN[1] and MEKA[2] repositories. The only exception is the "ces" dataset, described in [10].

The predictive performance of the algorithms was evaluated in terms of Accuracy, F-Measure, Hamming Loss and Exact Match. We used the well-known Friedman test and the Nemenyi post-hoc test to verify the statistical significance of the results at a confidence level of 95% [15]. As in the extensive evaluation of [20], a holdout evaluation was performed, where 2/3 of each dataset was used for learning the classifier and 1/3 for testing. We used the same training and test splits that come with the datasets.

As GACC and GA-PartCC algorithms are probabilistic

[1]http://mulan.sourceforge.net/datasets-mlc.html
[2]http://meka.sourceforge.net/#datasets

Table 1: Datasets used in the experiments

Dataset	q	N	Domain
birds	19	645	audio
cal500	174	502	music
enron	53	1702	text
genbase	27	662	genomics
llog	74	1,460	text
medical	45	978	text
thyroid	25	9,172	medical diagnosis
yeast	14	2,147	genomics
ces	17	903	social research

methods, the results reported are averaged over 10 executions with distinct seeds (except for the larger datasets "enron" and "llog" which were averaged over 5 executions). The results reported for CC are also averaged over 10 executions with distinct random seeds (5 executions for "enron" and "llog"). The parameters of the genetic algorithms were set by performing calibrating tests with three benchmark datasets obtained from the MULAN repository: emotions ($q = 6$, $N = 593$), scene ($q = 6$, $N = 2407$) and flags ($q = 7$, $N = 194$). These three datasets used for parameter calibration were not included in the experimental evaluation. The final settings are the following: number of generations: 50; population size: 200; crossover rate: 100%; mutation rate: 25%; tournament size: 2; elitist strategy: preserve the 2 best individuals at each generation. During the calibration tests, we observed the occurrence of overfitting, i.e., candidate solutions that have a high predictive accuracy on the training data, but do not generalize well for test instances [12]. To cope with this issue, we adopted the strategy of changing the building and validation sets at each generation of the GA by re-splitting the training set. It is worth reinforcing the fact that only the training set is used during the evolution process, as previously stated in Subsection 6.2.

7.2 Results

The results for the measures of Accuracy, F-Measure, Exact Match, and Hamming Loss are respectively shown in Tables 2, 3, 4 and 5. The best results for each dataset are highlighted in bold type. The rank obtained by each method in each dataset is presented in parenthesis. In the rows below Tables 2 and 3, the symbol $\succ$ represents a statistically significant difference between one or more methods. For instance, $\{a\} \succ \{b, c\}$ shows that the method a is significantly better than b and c.

The results presented in Tables 2 and 3 show that the GA-PartCC obtained the smallest rank sum (i.e., the best overall result) for both Accuracy and F-Measure. The Friedman test reported a significant difference between the evaluated methods. The Nemenyi post-hoc test indicated that GA-PartCC and GACC are significantly better than BR and CC for Accuracy and that GA-PartCC is significantly better than GACC, BR and CC for F-Measure.

Tables 4 and 5 show the results regarding the Exact Match and Hamming Loss measures. Although the Friedman and Nemenyi tests indicated that no statistically significant differences exist between the results of the four methods, note that the GA-PartCC obtained a competitive result, with the second smallest rank sum for both measures.

The GA-PartCC method was also evaluated with respect to the length of the chains associated to the best solutions. Table 6 shows the results averaged over the number of executions. Note that, for the datasets "llog" and "medical", the best models are on average composed by about half of the labels. For the other datasets, except "genbase", the best found models are, on average, composed by a large number of labels (but not all labels). This can be explained by the fact that in these datasets, a large number of labels exhibit dependence with each other. A notable exception is the dataset "genbase", in which the performance of the four evaluated methods is equivalent for all evaluation measures. In this case, the GA-PartCC method was consistent with the Occam's Razor principle, being able to choose the simplest model, an empty chain (equivalent to the BR model), as the best solution.

In summary, the results indicate that the proposed GA-PartCC method exhibits a very competitive performance, obtaining results significantly superior to the BR and CC methods according to two of the four evaluated measures of predictive accuracy. It was also significantly superior to GACC in one of the measures, and obtained competitive results in the other measures, yet with the advantage of generating simpler models than both CC and GACC.

8. CONCLUSIONS

This paper presented a novel method for multi-label classifier chains (CC) based on a genetic algorithm. This method, named GA-PartCC, differs from current extensions to the original CC model because it is capable of evaluating chain sequences that vary not only in the ordering but also in the length. In order to accomplish this task, GA-PartCC uses a variable-length list representation and a multi-objective lexicographic fitness function, taking into account two objectives: the model's accuracy and the model's size. In our experiments, GA-PartCC achieved statistically significantly better results than BR and CC in two of the four evaluated measures and a performance significantly superior to GACC in one of the measures, yet with the advantage of generating simpler models. None of the evaluated methods was significantly superior to GA-PartCC in any of the four performance measures.

As future research we plan to develop a memetic version of GA-PartCC. The idea is to identify, during the evolutionary process, groups of labels that tend to cause a decrease in the accuracy of most other labels when they take part into the chain. These labels could then be either removed from children resulting from crossover or placed in the last positions of these children. Similarly, groups of labels that tend to cause an increase in the predictive accuracy of other labels could be identified by the memetic procedure and placed in the first positions of new children.

9. ACKNOWLEDGMENTS

This work was supported by CAPES research grant BEX 1642/14-6 (E. C. Gonçalves).

10. REFERENCES

[1] M. P. Basgalupp, R. C. Barros, A. C. P. L. F. de Carvalho, A. A. Freitas, and D. D. Ruiz. Legal-tree: A lexicographic multi-objective genetic algorithm for decision tree induction. In *Proc. of the*

Table 2: Performance of BR, CC, GACC and GA-PartCC in terms of Accuracy.

Dataset	Accuracy			
	BR	CC	GACC	GA-PartCC
birds	**0.573** (1)	0.565 (4)	0.569 (2)	0.567 (3)
cal500	0.212 (4)	0.218 (3)	**0.224** (1)	0.222 (2)
enron	0.367 (4)	0.401 (3)	**0.409** (1)	0.405 (2)
genbase	**0.987** (2.5)	**0.987** (2.5)	**0.987** (2.5)	**0.987** (2.5)
llog	0.243 (2)	0.236 (4)	**0.249** (1)	0.242 (3)
medical	0.743 (3.5)	0.743 (3.5)	0.744 (2)	**0.748** (1)
thyroid	**0.984** (1.5)	0.983 (3.5)	0.983 (3.5)	**0.984** (1.5)
yeast	0.423 (3)	0.418 (4)	0.432 (2)	**0.440** (1)
ces	0.273 (4)	0.275 (3)	0.289 (2)	**0.292** (1)
rank sums	25.50	30.50	**17.00**	**17.00**

$\{GACC, GA\text{–}PartCC\} \succ \{BR, CC\}, and \{BR\} \succ \{CC\}$

Table 3: Performance of BR, CC, GACC and GA-PartCC in terms of F-Measure.

Dataset	F-Measure			
	BR	CC	GACC	GA-PartCC
birds	**0.603** (1)	0.592 (4)	0.597 (2)	0.595 (3)
cal500	0.344 (4)	0.348 (3)	**0.356** (1)	0.355 (2)
enron	0.474 (4)	0.504 (3)	**0.507** (1.5)	**0.507** (1.5)
genbase	**0.991** (2.5)	**0.991** (2.5)	**0.991** (2.5)	**0.991** (2.5)
llog	0.261 (3)	0.252 (4)	**0.264** (1.5)	**0.264** (1.5)
medical	**0.773** (1.5)	0.769 (3.5)	0.769 (3.5)	**0.773** (1.5)
thyroid	0.990 (3.5)	0.990 (3.5)	0.990 (3)	**0.991** (1)
yeast	0.547 (3)	0.523 (4)	0.548 (2)	**0.550** (1)
ces	0.372 (4)	0.373 (3)	0.393 (2)	**0.397** (1)
rank sums	26.00	30.00	19.00	**15.00**

$\{GA\text{–}PartCC\} \succ \{GACC, BR, CC\}, and \{GACC, BR\} \succ \{CC\}$

Table 4: Performance of BR, CC, GACC and GA-PartCC in terms of Exact Match.

Dataset	Exact Match			
	BR	CC	GACC	GA-PartCC
birds	0.486 (3)	0.485 (4)	**0.492** (1)	0.488 (2)
cal500	**0.000** (2.5)	**0.000** (2.5)	**0.000** (2.5)	**0.000** (2.5)
enron	0.086 (4)	0.125 (3)	**0.135** (1)	0.133 (2)
genbase	**0.975** (2.5)	**0.975** (2.5)	**0.975** (2.5)	**0.975** (2.5)
llog	0.199 (2)	0.198 (3.5)	**0.209** (1)	0.198 (3.5)
medical	0.651 (4)	0.665 (3)	0.670 (2)	**0.672** (1)
thyroid	0.903 (4)	0.941 (2.5)	0.941 (2.5)	**0.944** (1)
yeast	0.064 (4)	0.128 (3)	0.134 (2)	**0.136** (1)
ces	0.046 (3.5)	**0.049** (1)	0.047 (2)	0.046 (3.5)
rank sums	29.50	25.00	**16.50**	19.00

No statistically significant difference

Table 5: Performance of BR, CC, GACC and GA-PartCC in terms of Hamming Loss.

Dataset	Hamming Loss			
	BR	CC	GACC	GA-PartCC
birds	**0.051** (2)	**0.051** (2)	**0.051** (1)	0.052 (4)
cal500	**0.163** (1)	0.176 (4)	0.173 (4)	0.170 (2)
enron	**0.054** (1.5)	**0.054** (1.5)	0.055 (3.5)	0.055 (3.5)
genbase	**0.001** (2.5)	**0.001** (2.5)	**0.001** (2.5)	**0.001** (2.5)
llog	**0.019** (2.5)	**0.019** (2.5)	**0.019** (2.5)	**0.019** (2.5)
medical	**0.011** (2.5)	**0.011** (2.5)	**0.011** (2.5)	**0.011** (2.5)
thyroid	**0.005** (1.5)	0.006 (3.5)	0.006 (3.5)	**0.005** (1.5)
yeast	**0.259** (1)	0.274 (3)	0.266 (3)	0.263 (2)
ces	0.192 (4)	0.191 (3)	**0.190** (1.5)	**0.190** (1.5)
rank sums	**18.50**	25.50	24.00	22.00

No statistically significant difference

Table 6: Average length of the best chain determined by GA-PartCC

Dataset	q	avg length
birds	19	15.0
cal500	174	124.8
enron	53	52.0
genbase	27	0.0
llog	74	37.4
medical	45	20.2
thyroid	25	24.4
yeast	14	13.8
ces	17	13.7

2009 ACM Symposium on Applied Computing, SAC'09, pages 1085–1090, March 2009.

[2] L. Bull, E. Bernadó-Mansilla, and J. Holmes. Learning classifier systems in data mining: An introduction. In *Learning Classifier Systems in Data Mining*, volume 125 of *Studies in Computational Intelligence*, pages 1–15. Springer, 2008.

[3] P. N. da Silva, E. C. Gonçalves, A. Plastino, and A. A. Freitas. Distinct chains for different instances: An effective strategy for multi-label classifier chains. In *Proc. of the 2014 ECML/PKDD Conf., LCNS 8725*, pages 453–468, September 2014.

[4] P. Domingos. The role of occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery*, 3(4):409–425, December 1999.

[5] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. SpringerVerlag, 2003.

[6] A. A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002.

[7] A. A. Freitas. A critical review of multi-objective optimization in data mining: A position paper. *SIGKDD Explor. Newsl.*, 6(2):77–86, December 2004.

[8] A. A. Freitas. A review of evolutionary algorithms for data mining. In *Data Mining and Knowledge Discovery Handbook*, pages 371–400. Springer US, July 2010.

[9] A. A. Freitas. Comprehensible classification models: A position paper. *SIGKDD Explor. Newsl.*, 15(1):1–10, June 2013.

[10] E. C. Gonçalves. A human-centered approach for mining hybrid-dimensional association rules. In *Proc. of the 2014 Conf. on Information Fusion*, FUSION'14, pages 1–8, July 2014.

[11] E. C. Gonçalves, A. Plastino, and A. A. Freitas. A genetic algorithm for optimizing the label ordering in multi-label classifier chains. In *Proc. of the 25th Int'l Conf. on Tools with Artificial Intelligence*, ICTAI'13, pages 469–476, November 2013.

[12] I. Gonçalves and S. Silva. Balancing learning and overfitting in genetic programming with interleaved sampling of training data. In *Proc. of EuroGP 2013*, pages 73–84, April 2013.

[13] Y. Guo and S. Gu. Multi-label classification using conditional dependency networks. In *Proc. of the 22nd Int'l Joint Conf. on Artificial Intelligence*, IJCAI'11, pages 1300–1305, July 2011.

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The weka data mining software: an update. *ACM SIGKDD Explor.*, 11(1):10–18, November 2009.

[15] N. Japkowicz and M. Shah. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, New York, 2011.

[16] A. Kumar, S. Vembu, A. K. Menon, and C. Elkan. Beam search algorithms for multilabel learning. *Machine Learning*, 92(1):65–89, July 2013.

[17] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial Intelligence Review*, 13(2):129–170, April 1999.

[18] E. Levy, O. E. David, and N. S. Netanyahu. Genetic algorithms and deep learning for automatic painter classification. In *Proc. of the 2014 Genetic and Evolutionary Computation Conference*, GECCO'14, pages 1143–1150, July 2014.

[19] P. Machado, J. Tavares, F. B. Pereira, and E. Costa. Vehicle routing problem: Doing it the evolutionary way. In *Proc. of the 2002 Genetic and Evolutionary Computation Conference*, GECCO'02, pages 690–696, July 2002.

[20] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, September 2012.

[21] J. Read, L. Martino, and D. Luengo. Efficient monte carlo optimization for multi-label classifier chains. In *Proc. of the 38th Int'l Conf. on Acoustics, Speech, and Signal Processing*, ICASSP'13, May 2013.

[22] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, December 2011.

[23] A. E. Romero and L. M. de Campos. A probabilistic methodology for multilabel classification. *Intelligent Data Analysis*, 18(5):911–926, September 2014.

[24] L. E. Sucar, C. Bielza, E. F. Morales, P. H. Leal, J. H. Zaragoza, and P. Larrañaga. Multi-label classification with bayesian network-based chain classifiers. *Pattern Recogn. Lett.*, 41:14–22, May 2014.

[25] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010.

[26] G. Tsoumakas, E. S. Xioufis, J. Vilcek, and I. P. Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, July 2011.

[27] M. J. Zaki and W. Meira Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, New York, 2014.

[28] M.-L. Zhang and K. Zhang. Multi-label learning by exploiting label dependency. In *Proc. of the 16th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD'10, pages 999–1008, July 2010.

[29] M.-L. Zhang and Z.-H. Zhoug. A review on multi-label learning algorithms. *IEEE Trans. on Knowledge and Data Engineering*, 26(8):1819–1837, August 2014.