2013 IEEE 25th International Conference on Tools with Artificial Intelligence

A Genetic Algorithm for Optimizing the Label Ordering in Multi-Label Classifier Chains

Eduardo Corrêa Gonçalves, Alexandre Plastino
Instituto de Computação
Universidade Federal Fluminense (UFF)
Niterói, Brazil
{egoncalves,plastino}@ic.uff.br

Alex A. Freitas
Computing Laboratory
University of Kent at Canterbury
Canterbury, UK
A.A.Freitas@ukc.ac.uk

***Abstract*—First proposed in 2009, the classifier chains model (CC) has become one of the most influential algorithms for multi-label classification. It is distinguished by its simple and effective approach to exploit label dependencies. The CC method involves the training of q single-label binary classifiers, where each one is solely responsible for classifying a specific label in $\{l_1, \ldots, l_q\}$. These q classifiers are linked in a chain, such that each binary classifier is able to consider the labels predicted by the previous ones as additional information at classification time. The label ordering has a strong effect on predictive accuracy, however it is decided at random and/or combining random orders via an ensemble. A disadvantage of the ensemble approach consists of the fact that it is not suitable when the goal is to generate interpretable classifiers. To tackle this problem, in this work we propose a genetic algorithm for optimizing the label ordering in classifier chains. Experiments on diverse benchmark datasets, followed by the Wilcoxon test for assessing statistical significance, indicate that the proposed strategy produces more accurate classifiers.**

***Keywords*-multi-label classification; classifier chains; genetic algorithm**

I. INTRODUCTION

Multi-label classification (MLC) is the machine learning task of automatically assigning an object into multiple categories based on its characteristics [1], [2]. An example of practical application is music categorization, where the goal is to associate songs to music styles. For instance, most songs composed by Amy Winehouse can be classified as belonging to "Pop", "Jazz", and "R&B" styles. There are also many other important modern applications of MLC, such as functional genomics (determining the functions of genes and proteins), text categorization (associating documents to various subjects), medical diagnosis (patients might be suffering from more than one disease at the same time), direct marketing (recommendation of products for clients), etc.

The multi-label classification problem can be formally defined as follows. Let $X = R^d$ be the input feature space and $L = \{l_1, \ldots, l_q\}$ a set of q possible labels. Given a training set $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ $(x_i \in X, y_i \subseteq L)$, the goal of a multi-label classifier is to learn a function $h : X \rightarrow 2^L$ from D capable of predicting the labelset of an unseen instance. The MLC problem is more challenging than the traditional single-label classification (SLC) in which objects can be associated with only a single target class. This is due to two main reasons. The first is the fact that MLC applications often need to deal with an enormous number of possible label combinations. The size of the output space in MLC is 2^q whereas it is just q in SLC. The second concerns the existence of correlations between labels. For example, a song is unlikely to be simultaneously labeled as "Heavy Metal" and "Jazz" because these two music styles have a strong negative correlation. Thus, intuitively, we would expect that algorithms that are able to capture and model label correlations should be more accurate.

Actually, exploiting label dependencies has become a major concern in MLC research. A large body of recent work has primarily concentrated efforts to tackle this problem [3]–[11]. Proposed in [4], [5], the classifier chains model (CC) has become one of the most popular of such methods. This algorithm is mainly distinguished by its simple yet effective approach to take label correlations into account. The CC basic method consists of training q binary classifiers, one for each label. These classifiers are linked at random order in a chain $\{h_1 \rightarrow h_2 \rightarrow \ldots \rightarrow h_q\}$, such that the j^{th} classifier uses as input features not only the instance x, but also the output predictions of the previous $j - 1$ classifiers. While most MLC methods invest in complex probabilistic approaches to model label dependencies, the CC algorithm adopts a quite straightforward strategy: it just passes label information between classifiers. In spite of its simplicity, a comprehensive recent empirical study comparing several state-of-the-art methods for MLC [12] demonstrated that CC is among the top best performing algorithms in terms of predictive performance.

However, there is a drawback to CC, in that the label ordering is decided at random. It is evident that an inadequate label ordering can potentially decrease accuracy, as the first binary classifiers would frequently output wrong predictions at classification time, thus resulting in significant error propagation along the chain. In order to solve this issue, in [4], [5], the authors of CC propose combining random orders via an ensemble of classifier chains (ECC). In this strategy, the individual classifiers vote and the output labelset for a new instance is determined based on the collection votes. The expectation is that the effect of poorly ordered chains in predictive accuracy will be mitigated. Indeed, the machine learning literature have evidenced that in diverse SLC problems, ensembles are likely to be more accurate than their individual member classifiers [13]–[16].

1082-3409/13 $31.00 © 2013 IEEE
DOI 10.1109/ICTAI.2013.76

One of the main disadvantages of the ECC approach lies in the fact that it cannot be applied when the goal is to build interpretable classifiers. These kind of classifiers explain their classification decisions and are mainly represented by decision trees [17] and associative classifiers [18]. In some important application scenarios of MLC, such as medical diagnosis, bioinformatics and direct marketing, the ability to interpret the classification result might be almost as important as the accuracy itself. A second drawback associated to the ECC approach has been recently reported in [12]. This comprehensive study of MLC algorithms experimentally demonstrated that ECC exhibits a predictive accuracy inferior to CC on large datasets.

To cope with these problems, in this paper we propose the use of a genetic algorithm (GA) for optimizing the label ordering in a chain of classifiers. A genetic algorithm is a search technique based on natural evolution traditionally applied to solve classification [19] and optimization problems [20]. Our approach was mainly motivated by the following key aspects of GAs: (a) GAs perform a global search capable of effectively exploring the extremely large search space of $q!$ associated with optimizing the chain ordering; (b) GAs deliver an interpretable result, i.e., at the end of the process the algorithm will return a single optimized chain, reflecting the label dependencies.

The rest of this paper is organized as follows. Section 2 gives an overview of MLC concepts relevant to this paper. Section 3 addresses the CC and ECC models, discussing their main advantages and disadvantages. Our novel MLC method which employs GA is described in Section 4. We detail the experimental methodology and report experimental results on a set of benchmark datasets in Section 5. Section 6 revises the related work. Finally, we give concluding remarks and discuss some future research directions in Section 7.

II. BACKGROUND

In this section we give a brief overview of the basic approaches for mining multi-label classifiers with an emphasis on the binary relevance method. We also present the performance evaluation measures for MLC that are used in this paper. For a comprehensive introduction to these topics the reader is referred to [1], [2].

A. Basic Approaches for Multi-Label Learning

Existing methods for multi-label classification can be primarily divided into two main categories: algorithm dependent or independent.

Algorithm dependent methods extend or adapt an existing single-label algorithm for the task of multi-label classification. E.g., in [21] the authors modified the entropy formula and the structure of the leaves of the C4.5 [17] decision tree algorithm so as to allow its use in multi-label problems. Besides decision trees, other classic single-label techniques adapted for MLC include k-NN [3], SVM [11], and Bayesian networks [22].

On the other hand, algorithm independent methods transform the multi-label problem into one or more single-label problems. Then, any existing single-label algorithm can be directly applied by simply mapping its single label predictions into multi-label predictions. Algorithm independent methods are flexible, as they enable abstraction from the underlying base algorithm. This is an important advantage, because different classifiers achieve better performance in different application domains. There are a few different strategies to perform the transformation, with the binary relevance method (BR) [2], [4], [23] being the most commonly adopted in practice. In this approach, the multi-label problem is decomposed into q single-label binary problems. One binary classifier is independently trained for each label and new instances are predicted by combining the outputs produced by each classifier.

An example representing a hypothetical trained BR model for music categorization is illustrated in Figure 1. Observe that there are four labels involved in this problem: “Heavy Metal”, “Jazz”, “Blues”, and “Bossa Nova”. Consequently, four independent binary models had to be trained, being each one solely and exclusively responsible for classifying a specific label. The example illustrates the classification process of a new multi-label instance x (the song “Quiet Night of Quiet Stars”). To carry out the final classification, the BR model outputs the aggregation of the labels positively predicted by all the independent binary classifiers.

The BR strategy offers important advantages [4]. First, it is simple and algorithm independent. Second, it has relatively low computational complexity (scales linearly with q) and can be easily parallelized for better time performance. However, a considerable disadvantage lies in that each binary classifier works independently, ignoring the possible occurrence of relationships among labels. Section 3 discusses the classifier chains algorithm, an extension of the binary relevance method which exploits label dependencies.

B. Performance Evaluation

Over the last few years, several evaluation measures specifically designed for MLC have been proposed in literature. The platforms MULAN [24] and MEKA [25] - both widely adopted for research projects in MLC - make available more than twenty different metrics to their users. In this subsection, we introduce and compare the evaluation measures used in this paper. In the examples and definitions throughout the text we adopted the following notation:

- n : number of test instances.
- q : number of labels.
- Y_i: actual labelset of the i^{th} test instance.
- Z_i: predicted labelset of the i^{th} test instance.

The most trivial evaluation metric for MLC algorithms is the Exact Match (EM), defined in (1). This measure assesses the proportion of instances that were fully correctly predicted in the test set. Consider that I(true) = 1 and I(false) = 0.

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

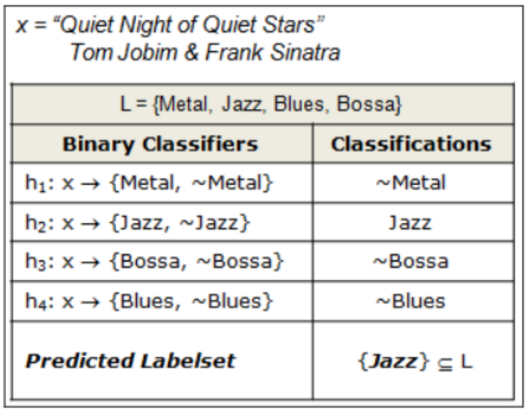

Figure 1. BR classification.

Although it provides essential information, the EM measure is regarded as too strict to be applied standalone. This is because in MLC problems a result can be, very often, partially correct, i.e., the classifier may predict some of the correct labels, but it can either miss some of them or include wrong predictions. Hence, it is necessary to adopt other evaluation metrics so as to complement the Exact Match.

The Accuracy measure (ACC), defined in (2), can be seen as a less harsh version of EM. It provides the user with information about the proportion of correct predictions, meanwhile taking into consideration results that are partially correct.

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

Table I illustrates the use of Accuracy on a toy problem of six test instances and four labels. Observe that the measure is simple and intuitive: the higher the value, the better the classification. However, the cases illustrated in the examples E_5 and E_6 highlight a drawback associated with Accuracy: the measure is not very severe on penalizing wrong predictions (either false positives or false negatives).

In this paper we also use the Hamming Loss measure (HL), defined in (3). This metric informs the average number of incorrect binary predictions per instance.

$$HL = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \,\Delta\, Z_i|}{q} \tag{3}$$

In (3), the expression $|Y_i \,\Delta\, Z_i|$ represents the symmetric difference between Y_i and Z_i. The use of HL on a new toy problem of six test instances and four labels is illustrated in Table II. Note that the most important advantage of HL over ACC is that it is more severe to penalize wrong predictions. Nevertheless, in many practical situations this characteristic ends up becoming a disadvantage. For instance, observe the four last examples in Table II. They show that, considering a pair of cases with the same number of wrong predictions (such as E_3 and E_4 or E_5 and E_6), the HL measure does not distinguish the case with a greater number of correct predictions from the one with a less number of correct predictions.

In summary, despite some idiosyncrasies, both Accuracy and Hamming Loss represent important performance evaluation measures that provide complementary information about MLC processes.

III. Chains of Classifiers

The BR method is a simple solution to the MLC problem, yet offering the advantages of being scalable to large datasets and algorithm independent. However, it has the serious disadvantage of ignoring the correlations between labels. This section is devoted to the classifier chains model (CC) [4], [5], an extension of BR which is able to exploit label correlations.

A. The Classifier Chains Model

As with BR, the classifier chains method involves the training of q single-label binary classifiers and each one will be solely responsible for classifying a specific label l_1, l_2, …, l_q. The difference is that, in CC, these q classifiers are linked in a chain $\{h_1 \rightarrow h_2 \rightarrow ... \rightarrow h_q\}$, such that, at classification time, each binary classifier h_j incorporates the labels predicted by the previous h_1, …, h_{j-1} classifiers as additional information. This is accomplished using a simple trick: in the training phase, the feature vector x for each classifier h_j is extended with the binary values of the labels l_1, …, l_{j-1}.

TABLE I. Illustration of Accuracy for Six Examples

	Yi	*Zi*	*ACC*	*Comments*
E_1	l_1, l_2, l_3, l_4	l_1, l_2, l_3, l_4	1.00	perfect classification
E_2	l_1	l_2, l_3, l_4	0.00	worst case
E_3	l_1	l_2	0.00	worst case
E_4	l_1, l_3	l_1, l_2	0.33	
E_5	l_1	l_1, l_2	0.50	
E_6	l_1, l_2	l_1, l_2, l_3, l_4	0.50	more false positives than E_5, but the value is still 0.50

TABLE II. Illustration of Hamming Loss for Six Examples

	Yi	*Zi*	*HL*	*Comments*
E_1	l_1, l_2, l_3, l_4	l_1, l_2, l_3, l_4	0.00	perfect classification
E_2	l_1	l_2, l_3, l_4	1.00	worst case
E_3	l_1	l_2	0.50	
E_4	l_1, l_3	l_1, l_2	0.50	better classification than E_3, but the value is still 0.50
E_5	l_1	l_1, l_2	0.25	
E_6	l_1, l_2, l_3, l_4	l_1, l_2, l_3	0.25	higher number of correct predictions than E_5, but the value is still 0.25

In Figure 2, we once again make use of the hypothetical example of music categorization, but this time illustrating the CC's classification process. In this example, consider the label ordering in the chain is assembled as: $\{h_{Metal} \rightarrow h_{Jazz}, \rightarrow h_{Bossa} \rightarrow h_{Blues}\}$. Differently from BR, the binary classifications of CC are not performed in a totally independent manner, since each classifier communicates its decision to the next elements in the chain. The classification process begins at h_1 and goes along the chain, i.e., the classifier h_j predicts the relevance of label j, given the feature space augmented by the predictions carried out by the previous $j - 1$ classifiers.

Despite its simplicity, the example in Figure 2 can highlight the advantage of allowing the communication among the binary models. Observe that a classifier such as h_3, which is placed near the end of the chain, can be clearly benefited from having been informed about the decisions of both h_1 and h_2. Thus, as shown in the example, it is able to come to a prediction different from the one generated by the BR model, illustrated in Figure 1.

B. *The Label Ordering Issue*

Although it employs a straightforward approach to exploiting label correlations, the CC algorithm is considered one of the most effective MLC methods, in the sense that it has proved to be competitive with state-of-the-art techniques [12]. Besides this, CC still maintains most of the attractive characteristics of BR, such as scalability and algorithm independence. Nevertheless, in the basic CC model, the label ordering is decided at random. This is a disadvantage, because an inadequate order may cause a significant decrease in predictive accuracy. It is intuitive that if the first members of the chain have low accuracy (i.e., if they output many wrong predictions), error propagation will occur along the chain. The authors of CC proposed the use of an ensemble of classifier chains (ECC) [4], [5] to mitigate the effect of poorly ordered chains. The motivation for their proposal lies in the fact that the machine learning literature have demonstrated that, in many problems associated to SLC, the use of ensembles lead to an increasing in accuracy [13]–[16].

However, there are two main drawbacks associated to ECC. First, ensembles are not suitable for applications that require comprehensible models. In some important applications of MLC, such as medical diagnosis, bioinformatics, and direct marketing, it is imperative to provide the users (doctors, biologists, managers, etc.) with means for interpreting the classification result. Second, a recent comprehensive comparison of MLC algorithms [12] experimentally demonstrated that ECC exhibits a predictive accuracy inferior to both CC and BR on large datasets. In fact, similar results had already been in reported in [5].

Motivated by these issues, in the next section we present the main contribution of this paper: a genetic algorithm for optimizing the label ordering in classifier chains.

IV. The Proposed Genetic Algorithm

A genetic algorithm (GA) is a search algorithm based on the Darwin's theory of natural evolution [19], [20]. In general, the GA search works as follows: in the first step, an initial population of individuals (also named chromosomes) is created, where each one corresponds to a candidate solution to a given problem. Next, these individuals are evaluated by a fitness function which assigns a numerical quality value to each of them. Then, the GA algorithm produces a new generation of individuals by employing the notion of "survival of the fittest". This procedure consists of selecting the best (fittest) individuals to be combined so as to produce a new generation resembling them (using genetic operators such as crossover and mutation). This process goes on for many iterations, progressively producing better and better candidate solutions. Normally, the GA execution terminates after a user-specified number of generations.

GAs have been widely employed to solve a large number of classification [19] and combinatorial optimization [20] problems, varying from social network mining [26] to financial analysis [27]. In this work, we propose the use of GA as a new approach for finding an optimized ordering for a chain of classifiers. In other words, the goal of the GA is to search for the label ordering that leads to an improvement on the predictive accuracy of the CC model.

x = "Quiet Night of Quiet Stars"
Tom Jobim & Frank Sinatra

L = {Metal, Jazz, Blues, Bossa}	
Binary Classifiers	**Classifications**
h_1: x → {Metal, ~Metal}	~Metal
h_2: x ∪ ~Metal → {Jazz, ~Jazz}	Jazz
h_3: x ∪ ~Metal ∪ Jazz → {Bossa, ~Bossa}	Bossa
h_4: x ∪ ~Metal ∪ Jazz ∪ Bossa → {Blues, ~Blues}	~Blues
Predicted Labelset	***{Jazz, Bossa}*** ⊆ L

Figure 2. CC classification.

Our motivations for developing GACC (GA for ordering Classifier Chains) were twofold: (a) GAs are a global search method, capable of effectively exploring the extremely large search space of $q!$ candidate solutions associated with the label ordering problem; (b) differently from ensembles, GAs can deliver an interpretable result (a single optimized chain ordering that can be interpreted by users). In Subsections 4-A, 4-B, and 4-C we present in details the designed GA.

A. Individual Encoding and Fitness Function

In the proposed approach, individuals of the population are simply represented by q-dimensional vectors regarding different specific label orderings for CC, i.e., each individual represents a permutation of the class labels.

The fitness function measures the quality of a candidate chain ordering. In our approach, it simultaneously takes into account the measures of Exact Match, Accuracy and Hamming Loss, respectively defined in Equations (1), (2) and (3). The fitness of an individual i is computed as:

$$Fitness(i) = \frac{EM + ACC + (1 - HL)}{3} \quad (4)$$

The proposed GA follows the wrapper approach [19], evaluating the quality of an individual (candidate label ordering) by using the target MLC algorithm (i.e. the CC algorithm). The fitness function is calculated using a holdout method as follows. First, the training set is partitioned into two mutually-exclusive subsets: building and validation. Then, we build a CC model using only the building set. Once the CC model is built, it is used to classify examples in the validation set.

B. Selection Method and Genetic Operators

At each generation, selection is performed using a tournament procedure [19]. First, the GA randomly chooses k individuals from the population, where k is a user-specified parameter called tournament size. These individuals "play a tournament" which consists of a comparison of their fitness values. The winner is the individual with the best fitness among the k participants.

Each pair of individuals selected by tournament undergo crossover operation in order to create offspring. This procedure is illustrated in Figure 3. First, a sub-chain is chosen at random on donor individual. Next, the elements of the sub-chain are removed from the receptor individual. At last, the sub-chain is inserted at random on the receptor.

The implemented mutation operation consists of selecting a single child from the new population and swapping two class labels (gene values in two different positions) at random in that child.

We also used elitism, in which two elite individuals are always preserved from the previous generation.

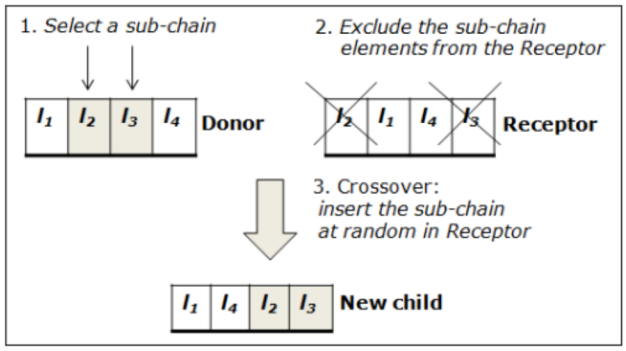

Figure 3. Crossover operation.

C. The Overfitting Issue

Since GAs perform a larger number of candidate solution evaluations (given by the number of individuals in the population times the number of generations), it is possible it will be prone to the problem of overfitting the training data - i.e., finding a classification model that achieves high accuracy on training data, but does not generalize well for unseen instances [28], [29]. In preliminary experiments, we observed the occurrence of overfitting when a large number of generations was set for the execution of the GA (e.g., 50 or 100 generations). To alleviate this problem and also to reduce the computational time required for the experiments, we followed the approach suggested in [30], limiting the number of generations to a value between 15 and 20, according to the size of the target dataset.

V. EXPERIMENTS

A. Datasets

The proposed genetic algorithm was evaluated on 10 different benchmark datasets. Their main characteristics are presented in Table III. The values in the second, third, and fourth columns (N, d, and q) represent the number of instances, features and labels, respectively. The values in the fifth column (*Lcard*) indicate the label cardinality (average number of labels per instance) [2]. The sixth column shows the application domain associated with each dataset. According to the complexity [5] of these datasets - which corresponds to the product Ndq - it is possible to categorize them into three groups: small ("Emotions", "Scene", "Flags", "Yeast", and "Cal500"), medium ("Medical", "Genbase", and "Enron") and large ("Corel5k" and "Bibtex").

All datasets were obtained from [24], except "Flags" which was obtained from the well-known UCI repository [31]. Most of them come divided into training and testing parts, being "Cal500" and "Flags" the only exceptions. Following the approach adopted in [12], we used these benchmark datasets with that predefined division, where the training part comprises about 2/3 of the complete dataset and the test part the remaining 1/3. For "Cal500" and "Flags" we generated the training and test parts.

TABLE III. MULTI-LABEL DATASETS USED IN THE EXPERIMENTS

Dataset	*N*	*d*	*q*	*LCard*	*Application Domain*
Emotions	593	72	6	1.87	Music
Scene	2407	296	6	1.07	Image
Flags	194	19	7	3.39	Image
Yeast	2417	103	14	2.16	Biology
Cal500	502	68	174	26.04	Music
Medical	978	1449	45	1.24	Medical Diagnosis
Genbase	662	1186	27	1.25	Biology
Enron	1702	1001	53	3.39	Text
Corel5k	5000	499	374	3.52	Image
Bibtex	7395	1836	159	2.40	Text

B. *Experimental Setup*

We implemented our GACC algorithm within the MULAN platform [24], a standard open source tool for the evaluation of multi-label algorithms that works on the top of the Weka framework [32]. The performance of GACC was compared to both Binary Relevance (BR) and Classifier Chains (CC) methods. J48 was used as the base SLC algorithm for these three methods. This algorithm consists of the Weka's implementation for the C4.5 [17], a well-known decision tree algorithm which produces interpretable classification models.

The parameters values used in GACC are shown in Table IV. Different settings were adopted according to the complexity of the target dataset. The identifiers *G*, *P*, and *k* represent the number of generations, the population size and the tournament size, respectively.

To assess whether the differences in performance are statistically significant we employed the Wilcoxon signed-Rank Test [33]. This is a non-parametric test appropriated for comparing two classifiers on multiple domains (datasets). It does not assume Normal distribution and works well for small sample sizes. In our case, each dataset corresponds to a data sample for the test (i.e., we have 10 data samples for the test), where the values being compared are the predictive performance measures of the classifiers. The significance of the results were verified with a confidence level of 95%.

C. *Results*

Tables V, VI and VII present the performance of each algorithm in terms of Exact Match, Accuracy and Hamming Loss, respectively. The best results for each dataset are highlighted in bold type.

TABLE IV. GACC PARAMETERS

Dataset size	**Parameters**		
	G	***P***	***k***
small	20	35	5
medium and large	15	20	3

According to Table V, the best values for Exact Match were achieved by GACC in the majority of the datasets. The two-tailed Wilcoxon test indicated that GACC is statistically superior to both BR and CC with a confidence level of 95% (W(7)=0 and W(7)=2, respectively).

The GACC method also outperformed the other models in terms of Accuracy in most datasets, as presented in Table VI. Once again, the statistical test suggested that the differences are significant with a confidence level of 95% (W(9)=2 and W(9)=5 in regard to BR and CC models, respectively).

The results presented in Table VII show that the BR model obtained the best results in terms of Hamming Loss. This was expected, since BR is actually suitable for most loss functions that ignore label correlations, as demonstrated in [6]. However, the differences between the Hamming Loss values of BR and GACC were very small on most of the datasets, except for "Emotions", the only dataset for which GACC outperformed both BR and CC. Actually, the two-tailed Wilcoxon test suggested that no statistically significant differences exist between the Hamming Loss values achieved by BR and GACC (W(9)=9) and also between the values achieved by CC and GACC (W(9)=17).

In summary, the results indicated that the proposed genetic algorithm obtained a gain on Exact Match and Accuracy without significantly impacting the Hamming Loss measure.

VI. RELATED WORK

This section briefly reviews the most recent work related to our proposal. In [7], the authors present the Bayesian Chain Classifier (BCC) algorithm, a technique that combines Bayesian networks with classifier chains. In this approach, the first step is to induce a Tree Augmented Network (TAN) [34] from the training dataset. This structure consists of a restricted Bayesian network in the form of a tree that represents correlations among labels. In the second step, an arbitrary node from TAN must be randomly chosen as the root node. From this root node, the different paths that compose the tree are used to form different chains. The Hybrid-Binary Chain Multi-Label Classifier (HCC), proposed in [8], employs a similar idea. In this approach, the correlation coefficient between each pair of labels is calculated in the first step. According to these results, different chains can be defined, each one composed by labels indentified as strongly positively correlated. Opposed to our approach that finds a single optimized chain ordering at the end of the GA evolution, neither BCC nor HCC generate a single specific chain. Instead, both algorithms first identify correlated labels and further employ this information to restrict the set of possible valid chain orderings.

Recently, two proposals that are actually targeted at determining a single suitable chain order have been introduced [9], [10]. The algorithm presented in [10], solves the label ordering problem by performing a beam search over a tree in which every distinct path represents a different label permutation. In order to avoid the construction of a tree with *q!* paths, an adjustable input parameter called *beam width* is employed in tandem with a technique based on kernel target

alignment score to prune uninteresting vertices. The M2CC algorithm, described in [9], relies on the use of a double Monte Carlo optimization technique to efficiently generate and evaluate a small population of distinct label orderings.

To the best of our knowledge, GACC is the first strategy that makes use of Evolutionary Algorithms to address the label ordering problem.

VII. CONCLUSIONS AND FUTURE WORK

Chain classifiers take label correlations into account and are still relatively efficient, exhibiting a good trade-off between accuracy and computational time. The label ordering has a strong effect on the predictive accuracy of these classifiers, nevertheless it is decided at random and/or combining random orders via an ensemble. In this paper we proposed GACC, a novel global method for optimizing the label ordering in CC that makes use of a genetic algorithm. Experiments on benchmark datasets indicated that, in general, GACC obtains results significantly superior to both BR and CC models, according to two of the three evaluated measures of predictive accuracy. Moreover, the GACC approach offers the advantage of being suitable for applications that require the generation of comprehensible classifiers, since it delivers to the user a single optimized chain ordering, reflecting the label dependencies.

For future research, we first plan to compare GACC against the four methods briefly introduced in Section VI. Also as future work, we intend to evaluate the effectiveness of different chromosome representations for GACC. For instance, we intend to define a structure capable of representing not only the chain ordering, but also the presence or the absence of any specific label. In this case, it will be possible to assess the predictive performance of shorter classifier chains (chains with length inferior to q). In addition, we leave as future work a detailed analysis on the sensivity of the results to GA parameters and other forms of implementing genetic operators. Finally, we plan to develop a parallel version of GACC to simultaneously process multiple candidate individuals.

TABLE V. PERFORMANCE OF EACH ALGORITHM IN TERMS OF EXACT MATCH

Dataset	*BR*	*CC*	*GACC*
Emotions	0.1287	0.1634	**0.2426**
Scene	0.4013	**0.5334**	0.5301
Flags	0.0769	**0.2000**	**0.2000**
Yeast	0.0643	0.1429	**0.1625**
Cal500	0.0000	0.0000	0.0000
Medical	0.6512	0.6822	**0.7116**
Genbase	**0.9794**	**0.9749**	**0.9749**
Enron	0.0864	0.1157	**0.1347**
Corel5k	**0.0020**	0.0000	**0.0020**
Bibtex	0.1332	0.1396	**0.1594**

TABLE VI. PERFORMANCE OF EACH ALGORITHM IN TERMS OF ACCURACY

Dataset	*BR*	*CC*	*GACC*
Emotions	0.4384	0.4277	**0.5076**
Scene	0.5134	**0.5945**	0.5858
Flags	**0.5763**	0.5587	0.5617
Yeast	0.4226	0.4317	**0.4552**
Cal500	0.2122	**0.2264**	0.2147
Medical	0.7426	0.7544	**0.7817**
Genbase	**0.9866**	**0.9866**	**0.9866**
Enron	0.3671	0.3941	**0.4092**
Corel5k	0.0753	0.0734	**0.0959**
Bibtex	0.2992	0.2879	**0.3250**

TABLE VII. PERFORMANCE OF EACH ALGORITHM IN TERMS OF HAMMING LOSS

Dataset	*BR*	*CC*	*GACC*
Emotions	0.2599	0.2896	**0.2129**
Scene	**0.1389**	0.1392	0.1465
Flags	**0.2747**	0.2989	0.2835
Yeast	**0.2588**	0.2638	0.2655
Cal500	**0.1631**	0.1729	0.1793
Medical	0.0106	**0.0103**	0.0108
Genbase	**0.0011**	**0.0011**	**0.0011**
Enron	0.0540	**0.0530**	0.0550
Corel5k	**0.0098**	0.0101	0.0114
Bibtex	**0.0148**	0.0149	0.0158

REFERENCES

[1] A. C. P. L. F. Carvalho and A. A. Freitas, "A Tutorial on Multi-label Classification Techniques," in *Studies in Computational Intelligence*, vol. 5, Springer, 2009, pp. 177–195.

[2] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining Multi-Label Data," in *Data Mining and Knowledge Discovery Handbook*, 2nd ed., Springer, 2010, pp. 667–685.

[3] M. L. Zhang and Z. H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, 2007.

[4] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier Chains for Multi-label Classification," in *Proc. of the ECML/PKDD 2009*, Bled, Slovenia, 2009, pp. 254–269.

[5] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier Chains for Multi-Label Classification," *Machine Learning*, vol. 85, no. 3, pp. 333–359, 2011.
[6] K. Dembczynski, W. Cheng, and E. Hullermeier, "Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains," in *Proc. of the 27th International Conference on Machine Learning*, Haifai, 2010, pp. 279–286.
[7] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, and P. Larrañaga, "Bayesian chain classifiers for multidimensional classification," in *IJCAI'11 Proc. of the 22nd international joint conference on Artificial Intelligence*, Barcelona, Spain, 2011, pp. 2192–2197.
[8] P. Hernandez-Leal, F. Orihuela-Espina, L. E. Sucar, and E. F. Morales, "Hybrid Binary-Chain Multi-label Classifiers," in *Proc. of the The 6th European Workshop on Probabilistic Graphical Models*, Granada, Spain, 2012.
[9] J. Read, L. Martino, and D. Luengo, "Efficient Monte Carlo Optimization for Multi-label Classifier Chains," in *Proc. of The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2013)*, Vancouver, Canada, 2013.
[10] A. Kumar, S. Vembu, A. K. Menon, and C. Elkan, "Beam search algorithms for multilabel learning," *Machine Learning*, vol. 92, no. 1, pp. 65–89, 2013.
[11] S. Godbole and S. Sarawagi, "Discriminative Methods for Multi-labeled Classification," in *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2004, pp. 22–30.
[12] G. Madjarov, D. Kocev, D. Gjorgjevikj, and D. Saso, "An Extensive Experimental Comparison of Methods for Multi-Label Learning," *Pattern Recognition*, vol. 45, 2012.
[13] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Addison-Wesley, 2005.
[14] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2011.
[15] G. Seni and J. F. Elder, *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*. Morgan and Claypool, 2010.
[16] L. Rokach, "Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography," *Computational Statistics & Data Analysis*, vol. 53, no. 12, pp. 4046–4072, 2009.
[17] J. R. Quinlan, *C4.5: Programs for machine learning*. Morgan Kaufmann Publishers, 1993.
[18] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," in *Proc. of the ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining*, New York, USA, 1998, pp. 80–86.
[19] A. A. Freitas, *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002.
[20] C. R. Reeves, "Genetic Algorithms," in *Handbook of Metaheuristics*, 2nd ed., Springer, 2010.
[21] A. J. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," *LNAI 2168*, 2001.
[22] C. Bielza, G. Li, and P. Larrañaga, "Multi-dimensional classification with Bayesian networks," *International Journal of Approximate Reasoning*, vol. 52, no. 6, pp. 705–727, 2011.
[23] T. Joachims, "Text Categorization with Suport Vector Machines: Learning with Many Relevant Features," in *Proc. of the 10th European Conf. on Machine Learning*, Germany, 1998, pp. 137–142.
[24] G. Tsoumakas, E. Spyromitros, J. Vilcek, and I. Vlahavas, "Mulan: A Java Library for Multi-Label Learning," *Journal of Machine Learning Research*, vol. 12, pp. 2411–2414, 2011.
[25] J. Read, *MEKA - A Multilabel/Multitarget Extension to WEKA*. 2013.
[26] D. Jin, D. He, D. Liu, and C. Baquero, "Genetic Algorithm with Local Search for Community Mining in Complex Networks," in *Proc. of the 2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, 2010.
[27] R. F. B. de Brito and A. L. I. Oliveira, "Comparative study of FOREX trading systems built with SVR+GHSOM and Genetic Algorithms optimization of technical indicators," in *Proc. of the 2012 24th IEEE International Conference on Tools with Artificial Intelligence*, 2012, pp. 351 – 358.
[28] L. A. Becker and M. Seshadri, "Comprehensibility and Overfitting Avoidance in Genetic Programming for Technical Trading Rules," Worcester Polytechnic Institute, Computer Science Technical Report WPI-CS- TR-03-09, 2003.
[29] I. Gonçalves and S. Silva, "Balancing Learning and Overfitting in Genetic Programming with Interleaved Sampling of Training Data," in *Proc. of EuroGP 2013*, Vienna, Austria, 2013, pp. 73–84.
[30] P. Domingos, "The Role of Occam's Razor in Knowledge Discovery," *Data Mining and Knowledge Discovery*, vol. 3, pp. 409–425, 1999.
[31] D. Newman, S. Hettich, C. Blake, and C. Merz, "UCI Repository of Machine Learning Databases.," 1998. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html.
[32] I. Witten, E. Frank, and H. Mark, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed. Morgan Kaufmann, 2011.
[33] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.
[34] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifiers," *Machine Learning*, vol. 29, 1997.