Multi-objective Genetic Programming for Multiple Instance Learning

Conference Paper · September 2007

DOI: 10.1007/978-3-540-74958-5_81

CITATIONS
7

READS
36

2 authors:

Amelia Zafra
University of Cordoba (Spain)
68 PUBLICATIONS **1,326** CITATIONS

SEE PROFILE

Sebastian Ventura
University of Cordoba (Spain)
342 PUBLICATIONS **12,196** CITATIONS

SEE PROFILE

Multi-objective Genetic Programming for Multiple Instance Learning

Amelia Zafra and Sebastián Ventura

Department of Computer Science and Numerical Analysis,
University of Córdoba
{azafra,sventura}@uco.es

Abstract. This paper introduces the use of multi-objective evolutionary algorithms in multiple instance learning. In order to achieve this purpose, a multi-objective grammar-guided genetic programming algorithm (MOG3P-MI) has been designed. This algorithm has been evaluated and compared to other existing multiple instance learning algorithms. Research on the performance of our algorithm is carried out on two well-known drug activity prediction problems, Musk and Mutagenesis, both problems being considered typical benchmarks in multiple instance problems. Computational experiments indicate that the application of the MOG3P-MI algorithm improves accuracy and decreases computational cost with respect to other techniques.

1 Introduction

Multiple instance learning, or multi-instance learning (MIL) introduced by Dietterich et al. [1] is a recent learning framework which has stirred interest in the machine learning community. In this paradigm, instances are organized in bags (i.e., multisets) and it is the bags, instead of individual instances, that are labeled for training. Multiple instance learners assume that every instance in a bag labeled negative is actually negative, whereas at least one instance in a bag labeled positive is actually positive. Note that a positive bag may contain negative instances.

Since its introduction, a wide range of tasks have been formulated as multi-instance problems. Among these tasks, we can cite content-based image retrieval [2] and annotation [3], text categorization [4], web index page recommendation [5,6] and drug activity prediction [7,8]. Also, a variety of algorithms have been introduced to learn in the multi-instance setting. Some of them are algorithms designed from scratch [1,7,8], while others [4,9,10,11,12,13,14] are based on well-known supervised learning algorithms. In this sense, the work of Zhou [15] is relevant in that it shows a general way in which supervised learners can be turned into multi-instance learners by shifting their focus from a discrimination on instances to the discrimination on the bags.

In this paper, we introduce a multi-objective grammar guided genetic programming algorithm designed to handle MIL problems. Our main motivations with this are: (a) genetic programming that allows a rule based classifier to be

J.N. Kok et al. (Eds.): ECML 2007, LNAI 4701, pp. 790–797, 2007.

generated (well known are the exceptional properties of these systems with respect to the comprehensibility and clarity of the knowledge being discovered) and (b) multi-objective strategy solutions that represent a tradeoff between different rule quality measurements, which is more interesting than maximising one individual measurement. As we will see, our algorithm (MOG3P-MI) generates a simple rule based classifier that increases generalization ability and includes interpretability and clarity in the knowledge discovered. Experiments are carried out by solving two well-known examples of drug activity prediction, Musk and Mutagenesis, which have been extensively used as benchmarks in evaluating and comparing MIL methods. Results show that this approach improves accuracy considerably with respect to existing techniques used to date.

The rest of this paper is organized as follows. Section 2 describes the proposed MOG3P-MI algorithm. Section 3 reports on experimental results. Finally, section 4 presents the conclusions and future work.

2 Multi-objective Genetic Programming for Multiple-Instance Learning

In this section we specify different aspects which have been taken into account in the design of the MOG3P-MI algorithm: individual representation, genetic operators and fitness function. With regard to the evolutionary process, our algorithm is based on the well-known Strength Pareto Evolutionary Algorithm 2 (SPEA2) [16] and, for this reason, no explanation about how it works is included.

2.1 Individual Representation

An individual consists of two components, a *genotype* which is encoded as tree structures with limitations in tree depth to avoid too large a size, and a *phenotype* which represents the full rule with antecedents and consequences. The antecedent consists of tree structures and represents a rule which can contain multiple comparisons attached by conjunction or disjunction, while the consequence specifies the class for the instance that satisfies all the conditions of the antecedent. To carry out the classification of the bags of instances, we use the formal definition of multi-instance coverage given by [15]. We consider that the possible value of the consequence is always the positive class, that is, all individuals are classified in the positive class and all examples that do not satisfy the individuals rule set are implicitly classified as belonging to the negative class.

We use a grammar to enforce syntactic constrains and satisfy the closure property (see Figure 1). This grammar mantains syntactical and semantic constraints in both the generation of individuals in the initial population and the production of new individuals via crossover.

2.2 Genetic Operators

The elements of the next population are generated by means of two operators: mutation and crossover.

```
antecedent ->          comparison
                     | OR comparison antecedent
                     | AND comparison antecedent

comparison ->        comparatorNumerical valuesToCompare
                     comparatorCategorical valuesToCompare

comparatorNumerical ->     <
                         | ≥

comparatorCategorical -> | CONTAIN
                         | NOT_CONTAIN

valuesToCompare ->  attribute  value
```

Fig. 1. Grammar used for individual representation

Mutation. The mutation operator can be applied to either a function node or a terminal node. A node in the tree is randomly selected. If the chosen node is a terminal it is simply replaced by another terminal. If it is a function, there are two possibilities with the same likelihood: (a) the function is replaced by a new function with the same arity and, (b) a new function node (not necessarily with the same arity) is chosen, and the original node together with its relative sub-tree is substituted by a new randomly generated sub-tree. If the new offspring is too large, it will be eliminated to avoid having invalid individuals.

Crossover. The crossover is performed by swapping the sub-trees of two parents between two compatible points randomly selected in each parent. Two tree nodes are compatible if their operators can be swapped without producing an invalid individual according to the defined grammar. If any of the two offspring is too large, they will be replaced by one of their parents.

2.3 Fitness Function

The fitness function evaluates the quality of each individual according to two indices that are normally used to evaluate the accuracy of algorithms in classification problems [17,18]. These are sensitivity and specificity. Sensitivity is the proportion of cases correctly identified as meeting a certain condition and specificity is the proportion of cases correctly identified as not meeting a certain condition. A value of 1 in both measures represents a perfect classification.

$$specificity = \frac{tn}{tn + fp}, \qquad sensitivity = \frac{tp}{tp + fn} \tag{1}$$

Where tp is the number of positive bags correctly predicted, tn is the number of negative bags correctly predicted, fp is the number of positive bags incorrectly predicted and fn is the number of negative bags incorrectly predicted.

Our fitness function combines these two indicators and the goal is to maximize them at same time. These measures are relationed, that is, there is a tradeoff

between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).

3 Experiments and Results

Our experiments are aimed to evaluate our proposed algorithm as compared to other classification techniques. Experimental results were estimated by 5 runs of 10-fold cross-validation with five different seeds for each partition and the average values of accuracy are reported in the next sections.

3.1 Datasets and Running Parameters

Experiments have been made on a drug activity prediction problem which is the most famous application for MIL. We discuss two datasets, Musk and Mutagenesis which are available at *http://www.cs.waikato.ac.nz/ml/milk*. The key properties of these datasets are shown in Table 1.

Table 1. Characteristics of the Musk and Mutagenesis datasets

	Musk		Mutagenesis	
Data Set	Musk1	Musk2	Easy	Hard
Number of bags	92	102	188	42
Number of positive bags	47	39	125	13
Number of negative bags	45	63	63	29
Number of instances	476	6598	10486	2132

These datasets are the most popular ones in the MIL domain, especially Musk. Every MIL algorithm developed so far has been tested using this problem. Therefore, we evaluated our algorithm based on these datasets.

The parameters used in all MOG3P-MI runs were: population size: 1000, external population size: 50, generations: 100, crossover probability: 95%, mutation probability: 15%, selection method for both parents: binary tournament selection with replacement, maximum tree depth: 15. The initial population was generated using the ramped-half-and-half method. The algorithm has been implemented in the JCLEC framework [19].

3.2 Comparison with Other Algorithms

Solving the Musk Problem. Comparisons are made with previous algorithms that include the ones specially designed for attacking the multiple-instance problem, as ITERATED-DISCRIM-APR which is the best of the four APR algorithms reported in [1], TILDE which is a top-down induction system for learning first order logical decision tree [20], CITATION-KNN [11] which is a variant of k nearest neighbour algorithm, RIPPER and RIPPERMI [21] which are a generic

Table 2. Summary results for Musk dataset

	Musk1	Musk2
Algorithm	Acc	Acc
MOG3P-MI	**0.93**	**0.93**
ITERATED-DISCRIM-APR	0.92	0.89
CITATION-KNN	0.92	0.86
DIVERSE DENSITY	0.89	0.82
RIPPERMI	0.88	0.77
NAIVE-RIPPERMI	0.88	0.77
TILDE	0.87	0.79
SVM	0.87	0.83

extension to propositional rule learners to handle multiple-instance data, Diverse Density [22] which is one of the most popular algorithms and MI-SVM [4] which is the best approach of support vectorial machines for MIL.

Table 2 shows a summary with the average values obtained by the different algorithms for each association of datasets. The results of the different algorithms are taken from [4] and [21].

In the Musk1 dataset, the hypotheses generated by MOG3P-MI contain an average of eight literals. These results are more accurate than those of other techniques which also generate interpretable knowledge. Moreover, the results are better than models which are not directly interpretable and have been specifically designed for this learning task, as ITERATED-DISCRIM-APR algorithm. The following is an example of rules generated by our algorithm.

IF (*(f10 > -220.3815)* $\wedge$ *(f7 > 35.2688)* $\wedge$ *(f163* $\leq$ *199.6827)*
$\wedge$ *(f55 > -84.9990)* $\wedge$ *(f134 > -216.0463)* $\wedge$ *(f34 > -216.1496)*
$\wedge$ *((f128* $\leq$ *18.1158)* $\vee$ *(f140 > 13.6820)* $\vee$ *(f136 > -78.2625))*)
THEN *Molecules have a musky smell.*
ELSE *Molecules have not a musky smell.*

In the Musk2 dataset, MOG3P-MI obtains an accuracy of 93%, which is far from the results found with the other techniques. This dataset has more bags and more instances by bags than in Musk1. However, our algorithm is not affected by these characteristics, and again it obtains the best results with respect to the rest of the techniques, these being comparable to those obtained in Musk1.

Solving the Mutagenesis Problem. The results of MOG3P-MI are compared to learners able to generate comprehensible hypotheses like PROGOL [23], FOIL and TILDE [20]. Also, we compare them to propositional rule learners, RIPPERMI [21] and NAIVE-RIPPERMI [21]. Table 3 displays the accuracy of MOG3P-MI, as well as the accuracy of five popular learners explained previously. The results of the different algorithms are taken from [21].

The best accuracy was obtained using Mutagenesis-hard which uses individual atoms and global molecular features that are highly correlated with the activity

Table 3. Summary results for Mutagenesis dataset

	Mutagenesis-easy	Mutagenesis-hard
Algorithm	Acc	Acc
MOG3P-MI	**0.84**	**1.00**
RIPPERMI	0.82	0.91
NAIVE-RIPPERMI	0.78	0.91
TILDE	0.77	0.86
PROGOL	0.76	0.86
FOIL	0.61	0.83

of the molecule. For this dataset, the other techniques obtain a good performance, but the best results are obtained by MOG3P-MI which obtains a perfect classification.

In a Mutagenesis-easy dataset, the accuracy of all the techniques fell. Nevertheless, our algorithm got the best results on the other ones. Thus, we can say that MOG3P-MI is competitive in terms of predictive accuracy, with respect to other learners. In addition, the induced hypotheses are concise, they contain an average of seven literals. The following is an example of the rule generated by our algorithm.

IF (*(element2 = 7.0)* $\wedge$ *(charge1 > -0.4862)* $\wedge$ *(charge2* $\leq$ *-0.5673)* $\wedge$
((quanta1 $\neq$ *9.0)* $\vee$ *(charge2 > -0.3719))*)
THEN *Molecules have mutagenic activity.*
ELSE *Molecules do not have a mutagenic activity.*

4 Conclusion and Future Work

The problem of MIL is a learning problem which has raised interest in the machine learning community. This problem is encountered in contexts where an object may have several alternative vectors to describe its different possible configurations.

In this paper, we describe the first attempt to apply multi-objective grammar guided genetic programming for multiple instance learning. MOG3P-MI is derived from the traditional G3P method and the SPEA2 multiobjective algorithm. Experiments on the Musk and Mutagenesis datasets show that our approach obtains the best results in terms of accuracy in the rest of the existing learning algorithm. Added to this are the benefits of interpretability and clarity in the knowledge discovered which provides a rule based system. The experimental study shows the success of our approach. However, further optimization is possible: issues such as the stopping criterion, the pruning strategy, and introduction of a third objective to improve the simplicity of obtaining easier rules would be an interesting issue for future work.

Acknowledgment

This work has been financed in part by the TIN2005-08386-C05-02 project of the Spanish Inter-Ministerial Commission of Science and Technology (CICYT) and FEDER funds.

References

1. Dietterich, T.G., Lathrop, R.H., Lozano-Perez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artifical Intelligence 89(1-2), 31–71 (1997)
2. Yang, C., Lozano-Perez, T.: Image database retrieval with multiple-instance learning techniques. In: ICDE '00: Proceedings of the 16th International Conference on Data Engineering, p. 233. IEEE Computer Society Press, Washington (2000)
3. Qi, X., Han, Y.: Incorporating multiple svms for automatic image annotation. Pattern Recognition 40(2), 728–741 (2007)
4. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: NIPS'02: Proceedings of Neural Information Processing System, pp. 561–568 (2002)
5. Zhou, Z.H., Jiang, K., Li, M.: Multi-instance learning based web mining. Applied Intelligence 22(2), 135–147 (2005)
6. Xue, X., Han, J., Jiang, Y., Zhou, Z.: Link recommendation in web index page based on multi-instance learning techniques. Jisuanji Yanjiu yu Fazhan/Computer Research and Development 44(3), 406–411 (2007)
7. Maron, O., Lozano-Pérez, T.: A framework for multiple-instance learning. In: NIPS'97: Proceedings of Neural Information Processing System 10, pp. 570–576. MIT Press, Cambridge (1997)
8. Zhang, Q., Goldman, S.: EM-DD: An improved multiple-instance learning technique. In: NIPS'01: Proceedings of Neural Information Processing System, pp. 1073–1080 (2001)
9. Zucker, J.D., Chevaleyre, Y.: Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. Application to the mutagenesis problem. In: Proceedings of the 14th Canadian Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence, Ottawa, Canada, pp. 204–214 (2000)
10. Ruffo, G.: Learning single and multiple instance decision tree for computer security applications. PhD thesis, Department of Computer Science. University of Turin, Torino, Italy (2000)
11. Wang, J., Zucker, J.D.: Solving the multiple-instance problem: A lazy learning approach. In: ICML'00: Proceedings of the Seventeenth International Conference on Machine Learning, pp. 1119–1126. Morgan Kaufmann Inc., San Francisco (2000)
12. Tao, Q., Scott, S., Vinodchandran, N.V., Osugi, T.T.: SVM-based generalized multiple-instance learning via approximate box counting. In: ICML'04: Proceedings of the twenty-first international conference on Machine learning, pp. 799–806. ACM Press, New York (2004)
13. Zhang, M.L., Zhou, Z.H.: Ensembles of multi-instance neural networks. In: Intelligent information processing II, vol. 163, pp. 471–474. Springer Boston, London, UK (2005)
14. Zhang, M.L., Zhou, Z.H.: Adapting RBF neural networks to multi-instance learning. Neural Processing Letters 23(1), 1–26 (2006)

15. Zhou, Z.H.: Multi-instance learning from supervised view. Journal Computer Science and Technology 21(5), 800–809 (2006)
16. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland (2001)
17. Bojarczuk, C.C., Lopes, H.S., Freitas, A.A.: Genetic programming for knowledge discovery in chest-pain diagnosis. IEEE Engineering in Medicine and Biology Magazine 19(4), 38–44 (2000)
18. Tan, K.C., Tay, Lee, A., Heng, T.H., C.M.: Mining multiple comprehensible classification rules using genetic programming. In: CEC'02: Proceedings of the Congress on Evolutionary Computation. Honolulu, HI, USA, vol. 2, pp. 1302–1307(2002)
19. Ventura, S., Romero, C., Zafra, A., Delgado, J.A., Hervás, C.: JCLEC: A java framework for evolutionary computation soft computing. Soft Computing (2007) ((in Press)
20. Blockeel, H., Raedt, L.D., Jacobs, N., Demoen, B.: Scaling up inductive logic programming by learning from interpretations. Data Mining Knowledge Discovery 3(1), 59–93 (1999)
21. Chevaleyre, Y., Zucker, J.D.: A framework for learning rules from multiple instance data. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 49–60. Springer, Heidelberg (2001)
22. Maron, O., Ratan, A.L.: Multiple-instance learning for natural scene classification. In: ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 341–349. Morgan Kaufmann Publishers Inc., San Francisco (1998)
23. Srinivasan, A., Muggleton, S.: Comparing the use of background knowledge by inductive logic programming systems. In: ILP '95: Proceedings of International Workshop on Inductive Logic Programming (1995)