Genetic programming as a feature selection algorithm

Conference Paper · November 2014

DOI: 10.1109/ROPEC.2014.7036345

3 authors, including:

José María Valencia-Ramírez
DEIPI - UMSNH
6 PUBLICATIONS **17** CITATIONS
SEE PROFILE

Mario Graff
Consejo Nacional de Ciencia y Tecnología
112 PUBLICATIONS **1,144** CITATIONS
SEE PROFILE

Genetic Programming as a Feature Selection Algorithm

Ranyart R. Suárez and José María Valencia-Ramírez and Mario Graff
Facultad de Ingeniería Eléctrica
Posgrado de Ingeniería Eléctrica
Universidad Michoacana de San Nicolás de Hidalgo
ranyart@dep.fie.umich.mx, jmvalencia@dep.fie.umich.mx, mgraffg@dep.fie.umich.mx

***Abstract*—Genetic Programming (GP) is an Evolutionary Algorithm commonly used to evolve computer programs in order to solve a particular task. Therefore, GP has been used to tackle different problems like classification and regression. In this work, the capabilities of GP in other types of problems are explored, particularly the feature selection problem. For this purpose, GP is applied to a set of benchmark problems, and, then, compared to other algorithms. The results obtained show that GP is competitive against the other algorithms, and in addition to this, no modifications are needed to perform the feature extraction task.**

I. Introduction

In the process of analysing a system, it is common practice to collect data that would provide useful information. In an ideal situation, only the data related to the problem would be collected; however, in real scenarios, normally one collects information no related to the system analysed and would mislead the understanding of the system.

That is, in most of the cases the data collected contains both representative information and noise. The problem of *feature selection* [1] consists in filtering the representative data contained in the data collected. Feature selection is needed to develop simpler models that consider only relevant features. There are many problems that require to make a feature selection step such as: classification, clustering, regression, feature learning, and online learning, among others.

Genetic Programming (GP) [2] is part of the *Evolutionary Algorithms (EAs)*, which are optimization algorithms based on the concepts of Evolution theory developed by Charles Darwin. Another examples of EAs are Genetic Algorithms (GAs), Differential Evolution (DE), Evolution Strategies (ES), Gene Expression Programming (GE), etc. These techniques are alternatives to traditional optimization algorithms and have proved to outperform them under some circumstances [3].

Like other EAs, GP creates an initial population of individuals, evaluate them with a *fitness function*, and then apply genetic operators like Crossover and Mutation to generate offspring than might be fitter than their parents. The difference between GP and other EAs is that in GP the individuals are computer programs which are commonly represented as trees. These trees are composed by functions and terminals, where the inner nodes are functions and the leafs are terminals. GP creates new individuals by randomly choosing between a set of functions and a set of terminals.

The objective of this work is to apply GP to feature selection problems, and, to experimentally show that GP can be competitive in this task when compared against other techniques which are specially designed for this purpose. It is important to note that GP is no tailor to work in this domain and a traditional GP system is used to perform this task.

The rest of this work is organized as follows: Section II presents the traditional algorithms used to perform feature selection. In Section III the methodology used to test GP as a feature selection is presented. Section IV presents the parameters of the GP systems used in this work, the parameters of the algorithms used to compare against GP, and information about the benchmark problems used in the comparison. Section V depicts the results obtained, and, finally, Section VI presents the conclusions and some research avenues for future work.

II. Related Work

In this work, GP is used to perform feature extraction over a set of benchmark problems. GP has been applied to feature selection problems in the past. In [4] GP was applied to Feature extraction, their procedure share some similarities with the current contribution, nonetheless there are significant difference as well. They propose a weight vector (generated in a previous filter step), which contains the weight of every feature in the problem, and then GP selects feature according to this vector. In this work there is also a filter step, but no weight vector is generated, instead, a subset of features is selected, and, then, in a second step, GP can select every feature with the same probability. Another difference is the benchmark problems, in [4] these problems have a small number of features (i.e., 6, 30 and 60 features), meanwhile, in this work, GP is tested on problems that contains several more features, for example, there is a problem with 10,000 features.

The results obtained in this work will be compared against two traditional algorithms for feature extraction: Random Forests, and, Regression Shrinkage and Selection (LASSO).

Random Forests (RF) [5] are ensemble of tree predictors [6], where the generalization error depends on the strength of individual trees in the forest and the correlation between them. RF have several desired qualities such as: RF are robust to outliers and noise, give useful internal information like error, strength, correlation, and, variable importance.

The variable importance computed by RF is the main advantage over other approaches for performing regression. This work uses the sklearn implementation of RF [7]. The variable importance computed by RF gives information about which explanatory variables are influencing the response variable in a greater degree. RF have been used for Feature Selection in the past [8]–[10].

LASSO [11] stands for *Least Absolute Shrinkage And Selection Operator*, is a technique which minimizes the sum of squared errors, where the sum of the absolute value of the coefficients is constrained to some criteria. LASSO can be seen as an improvement of Ordinary Least Squares (OLS). It also has similarities with another methods like forward stagewise regression, boosting and soft-thresholding of wavelet coefficients. LASSO has been used for Variable Selection in [12]–[14], Regression [15]–[17], and Model Selection [18].

III. GP for Feature Selection

Contrarily to the methods mentioned in Section II, GP is not related to feature selection problems. However, GP performs feature extraction *on the run* without any modifications to the algorithm. The reason for this is explained by the way GP selects the problem's variables. GP creates a population of computer's programs, these programs are created combining functions and terminals. The terminals are constants or problem's variables. These constants and variables are randomly selected every time GP creates a new individual. An individual may contain one or more variables more than once but it is unlikely in problems with several features that an individual contains all these features. Because of this, GP creates individuals that contain only a subset of the problem's variables.

In feature selection problems, the goal is to find a subset of variables that describes the best the behavior of the system. When GP is applied to feature selection problems, the creation of individuals would consider only a subset of features, and, once the evolution process has finished, the fittest individuals would contain only a subset of features. It can be inferred that this subset of features contains the features that affects in greater degree the outcome. This is why GP is performing a feature extraction on the run.

In order to prove that GP performs feature extraction without any changes in the algorithm, we decided to test GP over a set of benchmark problems corresponding to feature extraction. These problems were tested in an ensemble of GP systems. In machine learning, an ensemble is a combination of two or more prediction methods where the output of every method is weighted, and, then combined to produce the output of the ensemble. The main idea about creating an ensemble is to combined the advantages of various methods in order to increase the accuracy.

Like in any other EA, every time GP is used, a different result is obtained. This is why an ensemble of GP systems is used in this work, the results of each GP systems are combined in order to give a final result, which is the output of the ensemble. Typically, the size of the ensemble (number of predictors) is and odd number, in order to eliminate draws. In this work, the size of the ensemble used is 11.

Because GP is being used for feature extraction, two runs of the system are required, and, both of these runs are ensembles. In the first the most important features of the problem are identified. This step is accomplished by inspecting the fittest individuals in the ensemble. For example, if the ensemble is of size 11, 11 different GP systems are generated over the same problem. Each of these systems has a number of individuals that can be sorted by their performance. Therefore, n individuals can be selected from each GP system, that corresponds to the fittest individuals of that system. Finally, an histogram of features can be generated over these $n * 11$ individuals. From this histogram, the most important features for the GP system can be detected. The features with the highest number of occurrences, are the most important.

In the second stage the features selected are tested on the benchmark problems. That is, from the histogram of features, m features are selected, and only these features are considered in the second run. This subset containing the m features can be interpreted as the reduction of the irrelevant features of the problem. GP system performance in this second run must be competitive with the other feature extraction algorithms in order to prove that GP can be used to perform feature extraction.

In feature extraction, a common problem is to determine m, in other words, how many features consider as *relevant* and ignore the rest. All the algorithms try to minimize m without reducing the accuracy in the prediction. If the problem has many irrelevant features or noise, m can be approximately $10\% - 30\%$ of the total number of features. For the methodology proposed in this work, another consideration has to be made. This is the the value of n, which represents the number of individuals that are going to be considered in each ensemble member in order to build the features histogram. Each member of the ensemble is a GP system, which has a number of individuals sorted by their performance. Therefore, n represents the number of individuals that are chosen in each GP system.

Summarizing, the methodology proposed in this work consists of two steps: the filtering of the features, and the prediction of the classification problems. The GP systems used in these steps are both ensembles. From the first ensemble, the most relevant features are extracted. The second ensemble considers only the features extracted in the first step, and measures the performance of GP over the problem with the filtered features.

IV. Feature Selection Problems and Parameters of the Systems

GP, Random Forests and LASSO were tested over benchmark problems. These problems correspond to a contest in the area of feature selection which took place in the year 2003 (for the results of the competition see [19]). However, the system is still open for people who want to test their approach over the problems (*http://www.nipsfsc.ecs.soton.ac.uk/*).

The feature selection contests consisted of five different problems designed specially to test the feature selection methods. In this work, three problems from five were used to test the GP system and the other approaches. It is worth to mention that the missing two problems too large too handle in a reasonable amount of time for GP (these problems are 100,000 and 20,000 features large). The three problems used in this work correspond to a two class classification problem and are briefly explained below (we refer the interest reader to [20] for more information).

1) ARCENE (10,000 features) is a problem where the task is to identify prostate and ovarian cancer, the data was collected from two sources: The National Cancer Institute (NCI) and the Eastern Virginia Medical School (EVMS). The samples include patients with cancer and healthy or control patients.
2) GISETTE (5,000 features) the task in this problem is to classify two confusable handwritten digits: the four and the nine.
3) MADELON (500 features) is a problem where random data has to be classified. The samples in the dataset are synthetic.

Table I shows the GP parameters. We decided to use these parameters because in previous work these gave acceptable results (see [21]). In the case of the other algorithms RF and LASSO we decided to keep the *default* parameters available in the sklearn implementation [7].

TABLE I
GP PARAMETERS

Parameter	Value
Population Size	1000
Number of Generations	50
Function Set ($\mathcal{F}$)	$\{+, -, \times, /, \lvert\cdot\rvert, \exp, \sqrt{}; \sin, \cos, \text{sigmoid}, \text{if}, \max, \min, \ln, (\cdot)^2, \text{argmax}\}$
Crossover rate	90%
Mutation rate	10%
Mutation depth	random $\in [1, 5]$
Selection	Tournament of size 2

V. RESULTS

The results for the experiments are presented under two measures. The first one is simple the accuracy of the examples, and the second is the *Balanced Error Rate (BER)*, which is a statistical measure used to compare forecasters predictions. A definition of the BER measure can be found in [22]. First, the results of GP applied to the benchmarks problems are explained. Tables II, III and IV present the results for ARCENE, GISETTE, and MADELON problems, respectively. Such tables show the GP performance considering different percentage of features. There are different percentages of features for each problem because in the first step (the filter step), we realized that given the opportunity of choosing among all the variables, GP chose different percentages of features for all the three problems. GP chose $\approx 11\%$ of all the features for ARCENE, $\approx 21\%$ for GISETTE and $\approx 88\%$ for MADELON. So, we decided to perform the feature selection from 100% of features to 1% including these *fit* percentages. The best performance is shown in bold. The best performance of GP is a BER measure of 21.43 with 5% of features for ARCENE, BER measure of 6.70 with 21.3% of features for GISETTE and a BER measure of 21.33 with 5% of the features for MADELON. All these results correspond to tests set.

TABLE II
RESULTS OBTAINED BY GP USING A SUBSET OF FEATURES FOR TEST ARCENE'S TEST DATASET

% Features	Accuracy GP-E	BER GP-E
100.0%	73.00%	27.27
11.72%	75.00%	25.97
10.0%	73.00%	28.25
5.0%	**79.00%**	**21.43**
1.0%	73.00%	27.27

TABLE III
RESULTS OBTAINED BY GP USING A SUBSET OF FEATURES FOR TEST GISETTE'S TEST DATASET

% Features	Accuracy GP-E	BER GP-E
100.0%	92.00%	8.00
21.3%	**93.30%**	**6.70**
15.0%	92.20%	7.80
10.0%	92.50%	7.50
5.0%	91.60%	8.40
1.0%	91.50%	8.50

TABLE IV
RESULTS OBTAINED BY GP USING A SUBSET OF FEATURES FOR MADELON'S TEST DATASET

% Features	Accuracy GP-E	BER GP-E
100.0%	63.00%	37.00
88.4%	68.67%	31.33
50%	64.50%	35.50
40%	62.50%	37.50
30%	68.50%	31.50
20	68.50%	31.50
10%	74.50%	25.50
5%	**78.67%**	**21.33**
1%	67.67%	32.33

We decided to perform the exact same way with RF and LASSO, Tables V, VI and VII present the results for ARCENE, GISETTE and MADELON, in that order. The best results for RF were of 25.97 BER with 10% of features for ARCENE, 3.80 BER with 10% of features for GISETTE and 12.67 BER with 5% of features for MADELON. In the case of LASSO the best results were 20.05 BER with 8.76% of features for ARCENE, 2.90 BER with 12.9% of features for GISETTE and 40.16 BER with 1.45% of features for MADELON.

TABLE V
RESULTS OBTAINED BY TRADITIONAL METHODS USING A SUBSET OF FEATURES FOR TEST ARCENE'S TEST DATASET

-	RF		-	LASSO	
% Features	Accuracy	BER	% Features	Accuracy	BER
100.0%	74.00%	26.62	100.0%	65.00%	34.90
20%	74.00%	27.11	**8.76%**	**80.00%**	**20.05**
10%	**75.00%**	**25.97**	4.81%	77.00%	23.21
5.0%	72.00%	29.38	3.89%	75.00%	25.97
1.0%	74.00%	27.11	2.28%	77.00%	23.70

TABLE VI
RESULTS OBTAINED BY TRADITIONAL METHODS USING A SUBSET OF FEATURES FOR TEST GISETTE'S TEST DATASET

	RF			LASSO	
% Features	Accuracy	BER	% Features	Accuracy	BER
100.0%	95.30%	4.70	100.0%	85.60%	14.40
20%	95.60%	4.40	24.78%	96.90%	3.10
10%	**96.20%**	**3.80**	**12.9%**	**97.10%**	**2.90**
5.0%	95.70%	4.30	8.7%	96.80%	3.20
1.0%	94.00%	6.00	1.9%	96.40%	3.60

TABLE VII
RESULTS OBTAINED BY TRADITIONAL METHODS USING A SUBSET OF FEATURES FOR TEST MADELON'S TEST DATASET

	RF			LASSO	
% Features	Accuracy	BER	% Features	Accuracy	BER
100.0%	84.83%	15.17	100%	57.67%	42.33
20%	86.50%	13.50	17.12%	55.67%	44.33
10%	86.17%	13.83	10.56%	55.83%	44.16
5.0%	**87.33%**	**12.67**	3.90%	57.50%	42.50
1.0%	77.17%	22.83	**1.45%**	**59.83%**	**40.16**

How are these results compared to GP? Table VIII presents the best performance of each algorithm for al the three problems, the best performance is shown in bold for each case. From this table, we can see that GP performed second in the first case, in ARCENE the best was LASSO with an accuracy of 80% followed by GP with accuracy of 79%, a difference of only 1% and in third place RF with accuracy of 75%. For GISETTE the best was again LASSO with an accuracy of 97.10%, RF took the second place with accuracy of 96.20%, and, in third place, GP with an accuracy of 93.30. Lastly, in MADELON the best was RF with accuracy 87.33%, GP in second place with accuracy of 78.69%, and, in third place, LASSO with accuracy of 59.83.

Are these results bad for GP? we believe that they are not. The reason is because even though GP is not designed for feature extraction, it can perform the task with competitive results. For example, in ARCENE the difference with the first place, LASSO, is very little and moreover, GP selected fewer features than LASSO (5% against 8.76%). In MADELON the difference between GP and LASSO is very noticeable, BER measures of 21.33 for GP against 40.17 for LASSO. Table IX show the best performance of each algorithm in all the problems with the percentage of features that each algorithm selected. From this table, we can see that in the case of ARCENE, GP is selecting fewer variables that RF (5% against 10%) and has better accuracy.

TABLE VIII
BEST RESULTS OF GP COMPARED WITH TRADITIONAL METHODS FOR FEATURE SELECTION

	ARCENE		GISETTE		MADELON	
Method	Acc.	BER	Acc.	BER	Acc.	BER
GP-E	79.00%	21.43	93.30%	6.70	78.67%	21.33
RF	75.00%	25.97	96.20%	3.80	**87.33%**	**12.67**
LASSO	**80.00%**	**20.04**	**97.10%**	**2.90**	59.83%	40.17

TABLE IX
REDUCTION OF FEATURES BY DIFERENT METHODS

Method	ARCENE	GISETTE	MADELON
GP-E	**5.00%**	21.30%	5.00%
RF	10.00%	**10.00**	5.00%
LASSO	8.76%	12.90	**1.45%**

VI. CONCLUSIONS

The importance of feature selection is that it helps to filter the important features, therefore, saving computing time and storage space. At the same time it helps to increase the algorithm's performance because it removes features that do not provide problem's information, these non important features only provide noise, and this noise decreases the performance of the algorithm, like we can see in the results reported in this article.

The main objective of this work was to apply GP for feature selection problems, and then compare it to other algorithms like LASSO and RF which are specially designed for the task, and proof that is competitive. The results show that GP can outperform the other techniques under some problems. GP also selects fewer characteristics in some cases than the other algorithms, in other words, GP is reducing the noise of the benchmark problems better.

A. Future Work

We believe that if another parameters are chosen for GP, much better results can be obtained from the algorithm. The reason for this is because the parameters selected were tested on GP for wind prediction no feature selection. A further research of different parameters for GP can be easily implemented in order to try to improve the results of GP in Feature Selection problems.

REFERENCES

[1] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *AAAI*, pp. 129–134, 1992.

[2] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, "Genetic programming: An introduction: On the automatic evolution of computer programs and its applications (the morgan kaufmann series in artificial intelligence)," 1997.

[3] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary computation*, vol. 3, no. 1, pp. 1–16, 1995.

[4] A. Friedlander, K. Neshatian, and M. Zhang, "Meta-learning and feature ranking using genetic programming for classification: Variable terminal weighting," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pp. 941–948, IEEE, 2011.

[5] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[6] L. Breiman, *Classification and regression trees*. CRC press, 1993.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] Y. Saeys, T. Abeel, and Y. Van de Peer, "Robust feature selection using ensemble feature selection techniques," in *Machine learning and knowledge discovery in databases*, pp. 313–325, Springer, 2008.

[9] N. Chehata, L. Guo, and C. Mallet, "Airborne lidar feature selection for urban classification using random forests," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 39, no. Part 3/W8, pp. 207–12, 2009.

[10] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, "A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data," *BMC bioinformatics*, vol. 10, no. 1, p. 213, 2009.

[11] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[12] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the lasso," *The Annals of Statistics*, pp. 1436–1462, 2006.

[13] R. Tibshirani *et al.*, "The lasso method for variable selection in the cox model," *Statistics in medicine*, vol. 16, no. 4, pp. 385–395, 1997.

[14] I.-G. Chong and C.-H. Jun, "Performance of some variable selection methods when multicollinearity is present," *Chemometrics and Intelligent Laboratory Systems*, vol. 78, no. 1, pp. 103–112, 2005.

[15] J. O. Ogutu, T. Schulz-Streeck, and H.-P. Piepho, "Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions," in *BMC proceedings*, vol. 6, p. S10, BioMed Central Ltd, 2012.

[16] C.-H. Zhang and J. Huang, "The sparsity and bias of the lasso selection in high-dimensional linear regression," *The Annals of Statistics*, pp. 1567–1594, 2008.

[17] L. Meier, S. Van De Geer, and P. Bühlmann, "The group lasso for logistic regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 1, pp. 53–71, 2008.

[18] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[19] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the nips 2003 feature selection challenge," in *Advances in Neural Information Processing Systems*, pp. 545–552, 2004.

[20] I. Guyon, "Design of experiments of the nips 2003 variable selection benchmark," in *NIPS 2003 workshop on feature extraction and feature selection*, 2003.

[21] M. Graff, R. Pena, and A. Medina, "Wind speed forecasting using genetic programming," in *2013 IEEE Conference on Evolutionary Computation*, vol. 1, pp. 408–415, June 20-23 2013.

[22] Y.-W. Chen and C.-J. Lin, "Combining svms with various feature selection strategies," in *Feature extraction*, pp. 315–324, Springer, 2006.