Drawing Boundaries: Using Individual Evolved Class Boundaries for Binary Classification Problems

Jeannie Fitzgerald
BDS Group
CSIS Department
University of Limerick, Ireland
jeannie.fitzgerald@ul.ie

Conor Ryan
BDS Group
CSIS Department
University of Limerick, Ireland
conor.ryan@ul.ie

ABSTRACT

This paper describes a technique which can be used with Genetic Programming (GP) to reduce implicit bias in binary classification tasks. Arbitrarily chosen class boundaries can introduce bias, but if individuals can choose their *own* boundaries, tailored to their function set, then their outputs are automatically scaled into a suitable range. These boundaries evolve over time as the individuals adapt to the data

Our system calculates the *Evolved Class Boundary* (ECB) for each individual in every generation, with the twin aims of reducing training times and improving test fitness. The method is tested on three benchmark binary classification data sets from the medical domain.

The results obtained suggest that the strategy can improve training, validation and test fitness, and can also result in smaller individuals as well as reduced training times. Our approach is compared with a standard benchmark GP system, as well as with over *twenty* other systems from the literature, many of which use highly tuned, non-EC methods, and is shown to yield superior results in many cases.

Categories and Subject Descriptors

1.2.2 [**Artificial Intelligence**]: Automatic Programming - Program Modification

General Terms

Algorithms, Experimentation, Performance

Keywords

Genetic Programming, Binary Classification, Medical

1. INTRODUCTION

Classification problems arise in many application domains, including Internet search engines, document classification, credit scoring, image analysis, biometrical identification, and computer vision[26]. Accurate automated classification offers potential for significant benefits, and has long been an important area of study in Computer Science. Classification involves systematically assigning each of a given number of data items to a particular category. In machine learning terms, the data item is an instance represented by a feature vector describing attributes of that instance, the category to which the instance is assigned is referred to as a class, and the attribute that identifies to which class the instance is assigned is referred to as the class label. A class boundary is a decision point that determines the class label.

GP is suitable for classification tasks: it is domain independent, very flexible and expressive, and supports automatic feature detection and selection. The ability to automatically detect and select features offers potential for the development of fully automated classifiers. However, there are some drawbacks when applying GP to classification problems. The most notable of these are the long training times, difficulty in determining effective class boundaries for multi-class problems, and the lack of comprehensibility of the evolved solutions.[2, 14, 28].

To mitigate some of these problems, this paper proposes a simple technique, *Evolved Class Boundaries* (ECB), which makes it easier for the GP individuals to determine an appropriate class boundary. This is similar in operation to the Linear Scaling operation proposed by Keijzer [9] in that individuals are not penalized for not having access to ideal constants (either Ephemeral Random Constants or synthesized ones). It operates by automatically scaling the class boundary into the numerical range of the individual being tested, which permits the system to concentrate simply on distinguishing among the classes rather than having to evolve constants at the same time.

Our experiments suggest that this process can result in shorter training times and smaller programs. Since it is accepted [14, 19] that any multi-class problem of size n can be reduced to n binary classification tasks using a "one versus all" approach, it follows that the application of tools that facilitate the development of more effective binary classifiers can offer benefits in the multi-class situation.

The remainder of this paper is laid out as follows: Section 2 provides some background to this work with a particular focus on the application of class boundaries; Section 3 explains the proposed technique; Section 4 details the experimental set-up, results and comparison with other work, and Section 5 outlines conclusions and future work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'11, July 12–16, 2011, Dublin, Ireland.
Copyright 2011 ACM 978-1-4503-0557-0/11/07 ...$10.00.

2. BACKGROUND

GP has enjoyed much success with a wide variety of classification tasks [2, 13]. In a recent survey of work in which GP was applied to classification problems, Espejo et al [4] reviewed 66 papers where GP was compared with other methods with regard to classification accuracy. In 54.72% of cases GP was the best performing method.

A key step in the classification process is the determination of a class boundary. For binary classification tasks in GP this has traditionally been achieved in one of three ways:

Usually[21, 22], the boundary is set at zero, in which case the sign of an individual's output on a given training instance is used to determine the class label. Alternatively, the output of the program is itself a binary value which translates to a class label. For example, Bojarczuk et al [2] used logic operators to generate an if-then-else classification rule for chest pain diagnosis.

Another approach is to pass the output of individuals to a separate component whose function is the determination of the class label. Estebanez et al.[5] used a Linear Perceptron. Lam and Ciesielski [10] applied K-Means clustering to the program outputs, Smart and Zhang [19] proposed Gaussian distribution and probability models applied to program outputs. These last two approaches were applicable to both binary and multi-class classification tasks.

Several approaches have been recommended for multi-class problems using GP, such as Static Range Selection [27] and Dynamic Range Selection (DRS) [14]. The DRS algorithm used a subset of the training data set to devise population based class boundaries. Zhang and Smart [28] proposed two dynamic methods: Slotted Dynamic Range Selection(SDRS) and Centred Dynamic Range Selection(CDRS). The SDRS method employed "slots" in the range -25 to +25 and assigned the output of each program weighted by fitness to a particular slot, and each slot was assigned to a class. With CDRS, a centre for each class was calculated and the midpoints between centres formed the class boundary for the population as a whole. A similar approach using a decreasing weighted function was proposed by Li et al [11]. Each of the above methods were applied to image classification problems. Overall, it was found that Static Range Selection was effective with "easier" images but did not scale successfully. The dynamic methods performed well on more complex images but suffered from long training times and tended to produce large programs. Li et al.[11] applied parsimony pressure to mitigate the latter problem.

Several researchers have tackled the problem of multi-class classification in GP by decomposing the problem into multiple binary classification problems [21] [29]. The development of more effective binary classification algorithms could contribute towards achieving better solutions to multi class classification tasks.

3. DETAILS OF PROPOSED TECHNIQUE

There are some problems with the existing techniques used to determine class boundaries: the determination of good static boundaries may require hand crafted input by human experts [14], and the use of dynamic population based boundaries can result in long training times as the population strains to learn appropriate boundary values [20, 11].

With binary tasks, given feature vectors of positive real valued attributes, and a static class boundary of zero, individuals will take some time to move in the direction of the zero boundary. Similarly, given a population of very diverse individuals, a dynamic boundary that is designed for the population as a whole is likely to be challenging for individuals who differ from some mean population value. This is particularly likely to occur in the initial generations, and thus potentially useful genetic material may be discarded.

Rather than attempting to calculate boundaries that suit the entire population, our approach instead allows each individual to learn a class boundary that is natural for its genotype. We suggest that if the individual is permitted to chose a boundary that arises naturally out of the range of its outputs, then promising candidates may achieve good initial fitness and are less likely to be lost. In the proposed method, at each generation during training, the individual Evolved Class Boundary(ECB) is simply determined by calculating the mean of the individuals output for each class on the training data, and getting the midpoint of these two means.

$$boundaryP = \frac{\frac{\sum_i PO_i(C_1)}{mC_1} + \frac{\sum_j PO_j(C_2)}{mC_2}}{2} \qquad (1)$$

Where P represents an individual program, $PO_i(C_1)$ and $PO_j(C_2)$ are the outputs of the program for instances of Class 1 and Class 2, and mC_1 and mC_1 are the number of instances of Class 1 and Class 2 respectively. The final Evolved Class Boundary for each individual is the boundary that is applied to validation and test instances.

4. EXPERIMENTS

4.1 Data Sets

Three benchmark data sets from the medical domain have been used for experiments: The Wisconsin Breast Cancer data set, the Pima Indians Diabetes data set and the BUPA Liver Disorders data set.

4.1.1 Wisconsin Breast Cancer data set

The Wisconsin Breast Cancer (WBC) data set contains 699 instances of breast cancer diagnosis data from Wisconsin University Hospitals. Each consists of 10 integer valued attributes, including the instance identifier and class label, although sixteen of the instances have one or more attributes missing/unavailable. Those instances were discarded. Of the remaining, there are 458 instances of the "benign" class and 241 instances of the "malignant" class, and each of the training, validation and test sets were made up 224 instances of which 147 are benign instances and 77 malignant. Substantially more of the records with missing/unavailable attributes were of the benign class so it was necessary to also remove several of the others in order to preserve the ratio of positive to negative cases.

4.1.2 Pima Indians Diabetes data set

The Pima Indians have been the subject of intensive diabetes research due to the high incidence of the disease in the population, and the Pima Indians Diabetes (PID) data set consists of 768 instances taken from a larger database originally owned by the American National Institute of diabetes. The data in the PID data set refers to female patients at least twenty-one years old of Pima Indian heritage living in Phoenix, Arizona.

There are 500 negative cases and 268 positive cases in this dataset, each consisting of eight numerical attributes plus a class label. As with the WBC data, this data was evenly divided into training, validation and testing sets with 166 negative cases and 89 positive cases in each set.

4.1.3 *BUPA Liver Disorders data set*

The BUPA Liver Disorders(BUPA) data set, which contains 345 instances, refers to male patients and consists of information thought to be indicative of liver disorders. Each instance consists of seven numerical attributes including a class label, five attributes which are the results of blood tests that are thought to be sensitive to liver disorders and a final attribute which is the number of units of alcohol consumed daily. There are 200 positive instances and 145 negative instances and for our experiments we used 114 instances each for training, validation and test purposes, of which 66 are positive instances and 48 negative.

In partitioning the data between training, validation and test sets, we discarded several records from each master set where the values did not divide evenly, in order to ensure equal sized sets with the correct proportions, preserving the ratio of negative to positive instances. Copies of the data for these tasks were obtained from the UCI Machine Learning Database[6].

4.2 Function Sets

We have used a function set consisting of addition, subtraction, multiplication and protected division. It could be argued that the use of boolean operators would eliminate boundary bias and avoid the problem of developing dynamic boundaries. However, this approach may not scale well to multi-class tasks, and the application of strongly typed GP with mixed types may result in unnecessary complexity. More importantly, many classification problems have numeric rather than logical data, so it would seem more appropriate to use arithmetic operators in those cases. We have chosen not to use constants for our experiments, as the system should be capable of synthesising any values required using the function set and the range of values in the existing terminal sets.

Table 1: Common Parameters

Parameter	Value
Strategy	Generational
Initialisation	Ramped half-and-half
Selection	Tournament
Tournament Size	4
Crossover	70
Mutation	10
Elitism%	20
Initial Min Depth	1
Max Depth	12
Function Set	+ - * /
Population	500
Max Gen	50

Table 2: Specific Parameters

Parameter	WBC	PID	BUPA
# Terminals	9	8	5

4.3 Experimental Set-up and GP Parameters

Genetic Programming(GP) parameters used for the experiments are detailed in Tables 1 and 2. Each binary classification task was undertaken first using a baseline configuration, where the class boundary was set at zero, and the sign of the program output was used as the class label. Next, the experiments were repeated using the ECB configuration. Finally, experiments were run for the multi-class, population based “Centered Dynamic Class Boundary Determination”(CDCBD) method proposed by Smart and Zhang [28]. For each experiment, the data was divided into three sets: a training set, a validation set and a test set. The training set was used to train the GP Individuals on the particular experiment (PID,WBC or BUPA), while the validation set was employed to detect over fitting, and to select suitable candidates for testing; during training, each individual was evaluated against the validation set and a separate validation score was maintained (with no influence on training fitness). The individual with the highest validation score on each run was selected for evaluation on the corresponding test set. A useful application of a validation set is to curtail the training phase once some pre-defined validation criteria has been met, such as a deterioration or lack of improvement in validation fitness. On this occasion, we allowed the experiments to run to completion in order to obtain boundary behaviour data.

Fifty runs were undertaken for each configuration. For the static and ECB experiments, the runs were independent, whereas the CDCBD experiments were configured using the same random seeds as for the corresponding ECB runs. Overall, a total of 450 runs were completed. The fitness measure used for evolution was the number of classification errors of each program. Final fitness values are converted to error rate and % classification accuracy for reporting and comparison purposes. The GP framework used was the Open Beagle Evolutionary Framework[7]. For the remainder of this document the following naming conventions apply:

Static = Static Class Boundary (always set to zero)
CDCBD = Centered Dynamic Class Boundary[28]
ECB = Evolved Class Boundary

4.4 Results and Discussion

Results obtained using the baseline Static configuration are compared with those achieved using the CDCBD and ECB configurations and, outcomes for each of training, validation and test fitness, program size, and nodes processed are compared. In the case of training, the results displayed are the averages and standard deviation of the final generation of each run, whereas the values for validation and test results represent the average and standard deviation of the fifty selected validation individuals for each task when applied to the validation and test data respectively. We calculated the average number of nodes processed per run using the number of individuals processed and the average tree size for each generation, totalled over each run and then averaged over the fifty runs of each experiment. For each set of experiments, the best result in each category is displayed using bold text.

Table 3: WBC, PID & BUPA Training Validation & Test Fitness

	Training					Validation			Test		
Dataset	Method	Average Fitness	StdDev	Best Fitness	StdDev	Average Fitness	StdDev	Best Indiv.	Average Fitness	StdDev	Best Indiv
WBC	Static	94.95	0.74	98.44	0.41	96.97	0.73	98.28	95.58	1.10	97.41
	CDCBD	91.6	8.75	97.52	0.36	**97.50**	0.66	**98.66**	**97.42**	1.41	**99.55**
	ECB	**98.33**	0.51	**98.89**	0.2	96.75	0.28	97.41	96.42	0.54	97.41
PID	Static	72.27	2.27	77.03	1.73	70.91	1.23	74.5	67.52	3.07	74.9
	CDCBD	67.28	6.58	75.55	2.10	71.40	1.80	74.90	69.35	3.95	78.43
	ECB	**78.43**	1.34	**80.55**	0.68	**76.26**	0.61	**78.04**	**77.33**	1.33	**80.78**
BUPA	Static	76.04	3.03	81.39	2.84	72.60	2.49	77.19	71.95	4.15	78.94
	CDCBD	74.68	4.41	81.82	1.70	74.79	1.70	79.70	69.53	4.29	77.19
	ECB	**77.84**	2.08	**82.14**	1.8	**77.19**	2.04	**79.82**	**73.31**	2.32	**79.82**

4.4.1 *Training, Validation and Test Fitness Scores*

The results for the WBC experiments seen in Table 3 show that the ECB method produced better training results for both best and average fitness whereas the CDCBD method scored highest on validation and test fitness. Overall, both dynamic methods outperformed the static method. For the PID experiments, the ECB method consistently outscored the Static and CDCBD methods on each of the training, validation and test categories, and for average test fitness the result for the ECB configuration was almost 10% better than that of the Static configuration. Similarly, looking at the BUPA experiments, we see that the ECB approach produced better scores on each of training, validation and test fitnesses and the static method outperformed the CDCBD approach on test fitness.

The results in Table 3 demonstrate that the ECB technique delivered better results overall on the chosen tasks and both dynamic methods demonstrated superior performance over the static method, with the CDCBD approach doing best on the WBC problem and worst on the BUPA task. As the main objective when attempting to evolve classifiers is to generate ones that generalise well, one could argue that the most important statistic is average test fitness. In this respect, the ECB method has proven superior on two of the three experiments undertaken.

4.4.2 *Average Tree Size*

Results for average tree size shown in Table 4 represent the average tree size at generation 50 averaged over all runs for each task in the case of training. The tree sizes reported for validation/test are averages of the tree sizes of the fifty validation individuals which were selected for testing for each experiment. The results indicate that both the ECB and CDCBD methods produce significantly smaller trees overall, and between the two of these approaches, ECB produced smaller trees for two of the three tasks. Also, in each case, the average tree size of the validation/test individuals was smaller than the average of the population from which they were selected. This suggests that for this type of problem, smaller tress may generalise better. This is consistent with previous work of Nordin and Banzhaf[17] and the general principle of Occam's razor.

A small tree size is a desirable outcome, particularly when an uncomplicated function set is used, as the evolved results may be easier to interpret. Comprehensibility of the evolved solutions is important in certain problem domains, notably the medical domain, where the reasons for a particular classification may be as important as the classification itself. Similarly, in areas such as texture classification, it is useful to learn if the system has discovered discriminating features not previously understood.

Table 4: Average Tree Size

Data Set	**Method**	**Train**	**SdDev**	**Test**	**SdDev**
WBC	Static	136.3	81.44	112.92	87.70
	CDCBD	89.04	31.36	49.72	24.74
	ECB	**88.2**	32.88	**29.96**	19.3
PID	Static	241.25	190.67	176.06	213.8
	CDCBD	77.64	25.28	46.00	29.72
	ECB	**77.62**	30.7	**37.44**	28.28
BUPA	Static	480.76	666.34	346.08	625.4
	CDCBD	**113.04**	25.80	**94.24**	31.29
	ECB	163.98	146.37	106.2	147

Table 5: Average Nodes Processed

Data Set	**Method**	**Nodes (1000s)**
WBC	Static	2533
	CDCBD	1034
	ECB	**925**
PID	Static	2555
	CDCBD	1264
	ECB	**1170**
BUPA	Static	8569
	CDCBD	**1168**
	ECB	3412

4.4.3 *Average Nodes Processed per GP Run*

Table 5 shows the average number of nodes processed per GP run for each method. The results indicate that for the current experiments, significantly fewer nodes were processed during runs where the ECB and CDCBD methods were used. Comparing the ECB approach with the Static method, there were 63%, 55% and 60% fewer nodes processed for the WBC PID and BUPA tasks respectively when the technique was employed. The CDCBD technique used significantly fewer nodes for the BUPA task and generated results similar to the ECB approach for the other tasks. In GP, the most significant proportion of the computational cost is incurred in evaluating individual fitness. It follows that smaller individuals with fewer nodes to process should deliver shorter run times and use less memory overall.

As noted earlier, one of the perceived drawbacks to the application of GP to classification problems is the belief that it has a requirement for long training times when compared with other classification methods [14]. We suggest that the results for the current experiments demonstrate that use of the ECB technique can deliver dramatically reduced run times for this type of classification problem with virtually no trade off, as the boundary calculation is integrated with the normal evaluation process and does not involve any additional evaluations, which may be required when applying population based boundaries.

4.4.4 Statistical Tests

In order to ascertain whether the superior test results offered by the ECB method are significant, we carried out some simple statistical tests. Firstly, in order to check that the samples were normally distributed, we computed the z-score for the minimum and maximum values of each sample for each experimental task. Then, using a standard statistical hypothesis test [23] for large(> 30) independent samples, and choosing a significance level of 0.05, we calculated sample statistics for the differences in means of test fitness obtained for both the Static and ECB methods for each data set. The rationale in doing this is to establish whether it is likely that both samples are from the same or different populations, i.e if the ECB method has no significant effect then the means of both populations are equal (null hypothesis).

Test statistics of 4.86, 20.73 and 2.02 were obtained. The corresponding P-Values were 0.002, 0.002 and 0.0434. The P-Value represents the probability of getting a value of the sample test statistic that is at least as extreme as the one obtained from the sample data assuming the null hypothesis to be true. Using the significance level of 0.05, the ECB results can conventionally be described as statistically significant for the WBC, PID and BUPA experiments.

In comparing the performance of the CDCBD method against the ECB approach we used the Student t test for dependant samples. Test statistics of 11.71, 1.61 and -1 were obtained from the PID, BUPA and WBC data sets respectively. Using a significance level of 0.05 the critical values of t are -1.960 and 1.960. Thus, the difference in performance can be described as significant for the PID data, and not significant for the WBC and BUPA data.

4.4.5 ECB Behaviour

ECB works because individuals dictate their own boundary values. This section examines how the boundaries change over time. Due to the extremely wide range of the ECB values and the presence of significant numbers of valid outliers, we have chosen to report the median and median absolute deviation(MAD) for each generation rather than the average and standard deviation. The median is a robust measure of central tendency and the median absolute deviation is a robust measure of statistical dispersion. We consider that these robust measures present a more meaningful representation of the ECB data [8]. ECB values have been rounded to zero decimal places for reporting purposes only. Figures 1 and 2 show the behaviour of the ECB medians and median absolute deviation for a typical run of each problem. Median and Median absolute deviation values are scaled on the left axis in figures 1 and 2 respectively.

Observing the behaviour of the ECB across the three problems, we see that while the behaviour was noticeably different between the PID, BUPA and WBC problems, it was consistent across runs for each task. This would suggest that the system has used the boundaries in a problem specific way.

For the PID experiments, the boundaries typically ranged from very large (up to $\pm 10^{38}$) negative values to very large positive values early in the evolutionary process, and evolved towards large and very large positive values by the final generation.

Early evolution of the BUPA boundaries was similar to the PID boundary data where the individual boundaries ranged between very large negative and positive values. However as the process progressed, they moved closer to zero and at generation 50, generally ranged from small negative to positive values with a median close to zero.

The WBC experiments showed a different behaviour. In the first few generations, the underlying boundary data typically ranged from small negative to positive values with a small number of large positive outliers. At the final generation, these had generally evolved to being almost exclusively positive, in the range 0-500.

When comparing the Evolved Class Boundaries of the top and bottom 20% of the population at the final generation, it was clear that the boundary values of the fitter individuals ranged much closer together than those of their less fit counterparts. In the case of the PID experiments, the boundaries of the fitter individuals tended to be several degrees of magnitude smaller on average than those of the unfit portion of the population.

4.5 Comparison with other work

In this section, results obtained using the ECB approach are compared with other work using the same datasets, including non-EC approaches such as neural networks and SVMs. Results are expressed as either % classification accuracy or error rate as appropriate, to aid comparison with other work.

Lim et al. [12] compared the performance of over thirty classification algorithms including decision tree algorithms, statistical algorithms and neural networks. The best performing of these on the WBC, PID and BUPA datasets reported error rates of 0.03, 0.22 and 0.28 respectively. The corresponding best results (rounded) for ECB of 0.03, 0.23 and 0.27 on those problems are extremely competitive. However, in some cases, superior results to those of Lim et al have been reported elsewhere in the literature as detailed in Tables 6 to 8.

Looking firstly at comparative data for the Wisconsin Breast Cancer classification problem, we compare with Polat et al.[18] who detailed the performance of various algorithms on the task as shown in table 6. At 96.42% the ECB scores slightly below average when compared with the other results reported by Polat et al.

Table 6: WBC Comparative Data from Polat et al.[18]

Author (Year)	Method	%Acc.
Quinlan (1996)	C4.5	94.74
Hamilton et al. (1996)	RIAC	94.99
Nauck and Kruse (1999)	NEFCLASS	95.06
Current Work	Static	95.58
Abonyi and Szeifert (2003)	FuzzyClustering	95.57
Current Work	**ECB**	**96.42**
Ster and Dobnikar (1996)	LDA	96.80
Goodman et al. (2002)	Big-LVQ	96.80
Bennet and Blue (1997)	SVM	97.20
Goodman et al. (2002)	AIRS	97.20
Pena-Reyes andSipper (1999)	Fuzzy-GA1	97.36
Current Work	CDCBD	97.42
Setiono (2000)	Neuro-Rule 2a	98.10
Polat et al. (2005)	Fuzzy-AIRS	98.51

Table 7: Statlog[15] Data for PID Domain

Algorithm	Train	Test
Logdisc	0.219	0.223
Discrim	0.220	0.225
DIPOL92	0.220	0.224
ECB	**0.216**	**0.227**
SMART	0.177	0.232
RBF	0.218	0.243
ITrule	0.223	0.245
Backprop	0.198	0.248
Cal5	0.232	0.250
CART	0.227	0.255
CASTLE	0.260	0.258
NaiveBay	0.239	0.262
Quadisc	0.237	0.262
C4.5	0.131	0.270
IndCART	0.079	0.271
Baytree	0.008	0.271
LVQ	0.101	0.272
Kohonen	0.134	0.273
Kohonen	0.134	0.273
AC	0.00	0.276
CN2	0.010	0.289
NewID	0.00	0.289
ALLOC80	0.288	0.301
CDCBD	0.327	0.306
k-NN	0.000	0.324
Static	0.277	0.325
Default	0.350	0.350

For the PID classification task we compare results with those reported in Statlog as shown in Table 7[15]. Here, the rank associated with each algorithm is based on test classification error. Using this criteria, the ECB configuration ranks fourth on the StatLog scale with an error rate of 0.227. In more recent work by Eggermont et al. [3] several GP variants were tested. The best of these had an error rate of 0.242. Tsakonas [24] tested four GP algorithms and reported an error rate of 0.2198 for GP with Fuzzy Rule Based Systems. Considering these results, the performance offered by the ECB technique could be described as very competitive.

Referring once again to the work of Polat et al.[18] in Table 8 we make a comparison for the BUPA Liver Disorders task. Here, the ECB configuration resulted in a classification accuracy of 73.31% which scores fourth highest of the methods listed and is only outscored by RSVM, AIRS and Fuzzy-AIRS. The Static method also performs respectably with accuracy of 71.95%. In other work. Loveard and Ciesielski [14] experimented with GP using static and population based dynamic boundaries. They recorded a best test score of 69.2%. Muni et al. [16] reported a similar result. Recent research by Badhran and Rockett(2010) [1] applied GP with varying parameters and reported a test result of 74.86%.

Table 8: BUPA Comparative Data from Polat et al.[18]

Author (year)	Method	%Acc.
Polat et al. (2005)	Fuzzy-AIRS	83.38
Polat et al. (2005)	AIRS	81.00
Lee and Mangasarian (2001b)	RSVM	74.86
Current work	**ECB**	**73.31**
Yalçın and Yıldırım (2003)	MLP	73.05
Current work	Static	71.95
Lee and Mangasarian (2001a)	SSVM	70.33
Van Gestel et al. (2002)	SVM with GP	69.70
Current work	CDCBD	69.53
Cheung (2001)	C4.5	65.59
Yalçın and Yıldırım (2003)	GRNN	65.55
Cheung (2001)	Naive Bayes	63.39
Cheung (2001)	BNND	61.83

In summary, we believe that the results and comparisons show that the Evolved Class Boundary method has delivered a competitive performance for the Wisconsin Breast Cancer data, with competitive and often superior performances for both the Pima Indians Diabetes and BUPA Liver Disorders classification tasks.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a technique that can be used with GP for to improve performance on binary classification tasks. We believe that we have shown that the proposed method, ECB, can improve training, validation and test results, reduce run times and produce smaller trees. Although the method is extremely simple, it has proven to be surprisingly effective. Most importantly, the technique offered good results on test data, and we have established that these results are likely to be statistically significant. In comparison with other work on the same problems, the proposed method has delivered competitive and sometimes superior results.

The data sets used in our experiments do not exhibit a high degree of class imbalance, having an approximate ratio of 2:1 of positive to negative instances or vice versa. Should future work involve the use of data sets with higher degrees

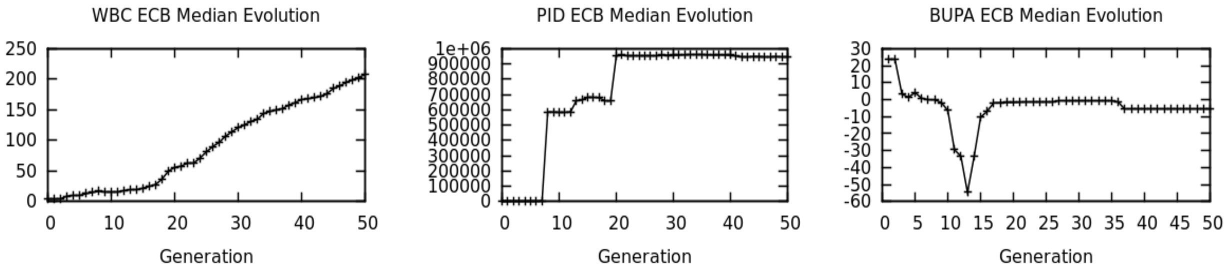

Figure 1: Evolution of ECB Median values for typical PID, BUPA and WBC runs, Median on left axis

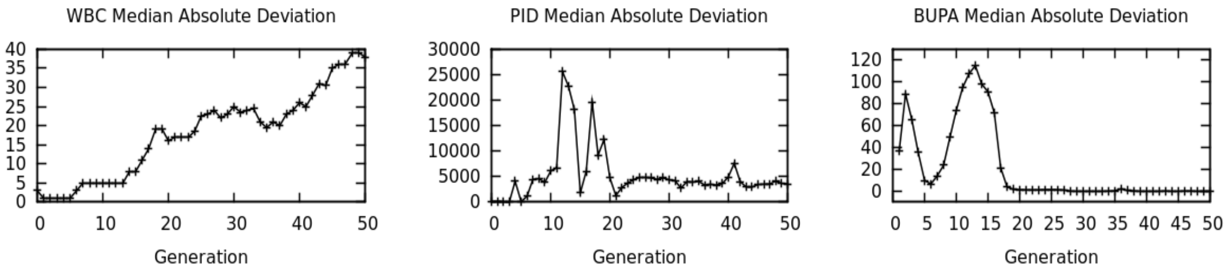

Figure 2: Evolution of ECB Median Absolute Deviation values for typical BUPA, PID and WBC runs, MAD on left axis

of class imbalance, this would need to be taken into account. Similarly, we have used a very simple function set, and can reasonably conclude that this, combined with the relatively small trees generated by the ECB method implies a low level of functional complexity. Should more complex functions be used in future work, it would be necessary to undertake more detailed analysis of the chosen individuals by employing for example some of the ideas suggested by Vladislavleva et al.[25].

Revisiting the current work, it would be interesting to examine the way in which the system seems to be able to utilise the boundaries in a problem specific way. Also, we would like to learn if the ECB technique can be successfully applied to more general classification problems. By taking the current work as an initial step we plan to investigate the potential of the technique to solve multi-class problems by binary decomposition.

6. REFERENCES

[1] K. Badran and P. I. Rockett. The influence of mutation on population dynamics in multiobjective genetic programming. *Genetic Programming and Evolvable Machines*, 11(1):5–33, Mar. 2010.

[2] C. C. Bojarczuk, H. S. Lopes, and A. A. Freitas. Discovering comprehensible classification rules by using genetic programming: a case study in a medical domain. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 953–958, Orlando, Florida, USA, 13-17 July 1999. Morgan Kaufmann.

[3] J. Eggermont, J. N. Kok, and W. A. Kosters. Genetic programming for data classification: Partitioning the search space. In *Proceedings of the 2004 Symposium on Applied Computing (ACM SAC'04)*, pages 1001–1005, Nicosia, Cyprus, 14-17 Mar. 2004.

[4] P. G. Espejo, S. Ventura, and F. Herrera. A Survey on the Application of Genetic Programming to Classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(2):121–144, March 2010.

[5] C. Estebanez, R. Aler, and J. M. Valls. A method based on genetic programming for improving the quality of datasets in classification problems. *International Journal of Computer Science and Applications*, 4(1):69–80, 2007.

[6] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[7] C. Gagné and M. Parizeau. Open beagle: A new c++ evolutionary computation framework. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '02, pages 888–, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[8] D. C. Hoaglin, F. Mosteller, and J. W. Tukey, editors. *Understanding robust and exploratory data analysis.* Wiley series in probability and mathematical statistics. Wiley-Interscience, 1983.

[9] M. Keijzer. Scaled symbolic regression. *Genetic Programming and Evolvable Machines*, 5:259–269, September 2004.

[10] B. Lam and V. Ciesielski. Discovery of human-competitive image texture feature extraction using genetic programming. In K. D. et al., editor,

LNCS, volume 3103, pages 1114–1125. GECCO, Springer-Verlag Berlin Heidelberg, 2004.

[11] Y. Li, J. Ma, and Q. Zhao. Two improvements in genetic programming for image classification. In J. Wang, editor, *2008 IEEE World Congress on Computational Intelligence*, pages 2492–2497, Hong Kong, 1-6 June 2008. IEEE Computational Intelligence Society, IEEE Press.

[12] T.-S. Lim, W.-Y. LOH, and W. Cohen. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, 2000.

[13] M. Lotz and S. Silva. Application of genetic programming classification in an industrial process resulting in greenhouse gas emission reductions. In C. Di Chio, A. Brabazon, G. A. Di Caro, M. Ebner, M. Farooq, A. Fink, J. Grahl, G. Greenfield, P. Machado, M. O'Neill, E. Tarantino, and N. Urquhart, editors, *EvoENVIRONMENT*, volume 6025 of *LNCS*, pages 131–140, Istanbul, 7-9 Apr. 2010. Springer.

[14] T. Loveard and V. Ciesielski. Representing classification problems in genetic programming. In *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1070–1077, COEX, Seoul, Korea, 27-30 May 2001. IEEE Press.

[15] D. Michie, D. J. Spiegelhalter, C. C. Taylor, and J. Campbell, editors. *Machine learning, neural and statistical classification.* Ellis Horwood, Upper Saddle River, NJ, USA, 1994.

[16] D. P. Muni, N. R. Pal, and J. Das. A novel approach to design classifier using genetic programming. *IEEE Transactions on Evolutionary Computation*, 8(2):183–196, Apr. 2004.

[17] P. Nordin and W. Banzhaf. Complexity compression and evolution. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 310–317, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[18] K. Polat and S. Güneş. Artificial immune recognition system with fuzzy resource allocation mechanism classifier, principal component analysis and fft method based new hybrid automated identification system for classification of eeg signals. *Expert Syst. Appl.*, 34:2039–2048, April 2008.

[19] W. Smart and M. Zhang. Using genetic programming for multiclass classification by simultaneously solving component binary classification problems. Technical Report CS-TR-05-1, Computer Science, Victoria University of Wellington, New Zealand, 2005.

[20] W. R. Smart and M. Zhang. Classification strategies for image classification in genetic programming. In D. Bailey, editor, *Proceeding of Image and Vision Computing NZ International Conference*, pages 402–407, Palmerston North, New Zealand, Nov. 2003. Massey University.

[21] A. Song, T. Loveard, and V. Ciesielski. Towards genetic programming for texture classification. In M. Stumptner, D. Corbett, and M. Brooks, editors, *Proceedings of the 14th International Joint Conference on Artificial Intelligence AI 2001: Advances in Artificial Intelligence*, volume 2256 of *Lecture Notes in Computer Science*, pages 461–472, Adelaide, Australia, Dec. 10-14 2001. Springer-Verlag.

[22] W. A. Tackett. Genetic programming for feature discovery and image discrimination. In S. Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pages 303–309, University of Illinois at Urbana-Champaign, 17-21 July 1993. Morgan Kaufmann.

[23] M. F. Triola. *Elementary Statistics.* Addison-Wesley, 6 edition, 1995.

[24] A. Tsakonas. A comparison of classification accuracy of four genetic programming-evolved intelligent structures. *Inf. Sci.*, 176:691–724, March 2006.

[25] E. J. Vladislavleva, G. F. Smits, and D. Den Hertog. Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming. *Trans. Evol. Comp*, 13:333–349, April 2009.

[26] C. Weihs and W. Gaul, editors. *Classification - the Ubiquitous Challenge, Proceedings of the 28th Annual Conference of the Gesellschaft für Klassifikation e.V., University of Dortmund, March 9-11, 2004*, Studies in Classification, Data Analysis, and Knowledge Organization. Springer, 2005.

[27] M. Zhang and V. Ciesielski. Genetic programming for multiple class object detection. In N. Foo, editor, *12th Australian Joint Conference on Artificial Intelligence*, volume 1747 of *LNAI*, pages 180–192, Sydney, Australia, 6-10 Dec. 1999. Springer-Verlag.

[28] M. Zhang and W. Smart. Multiclass object classification using genetic programming. Technical Report CS-TR-04-2, Computer Science, Victoria University of Wellington, New Zealand, 2004.

[29] C. Zhou, P. C. Nelson, W. Xiao, and T. M. Tirpak. Discovery of classification rules by using gene expression programming. In *Proceedings of the International Conference on Artificial Intelligence 2002*, pages 1355–1361, Las Vegas, U.S.A., June 2002.