Genetic Programming Approach to Benelearn 99: II

W.B. Langdon[1]

Centrum voor Wiskunde en Informatica, Kruislaan 413, NL-1098 SJ, Amsterdam
bill@cwi.nl http://www.cwi.nl/~bill
Tel: +31 20 592 4093, Fax: +31 20 592 4199

Abstract. We briefly describe our first genetic programming technique used to automatically evolve profiles of potential insurance customers the task is part of the Benelearn 99 competition. The information about customers consists of 86 variables and includes product usage data and socio-demographic data derived from zip codes. The data was supplied by the Dutch data mining company Sentient Machine Research, and is based on real world business data. Profiles which correctly identified more than 50% of customers were automatically evolved using genetic programming.
Models are completely automatically generated by GP 1) starting from random and 2) starting from C4.5 and improving on it. The models evolved are similiar in performance in the two cases.

1 Genetic Programming and Pareto Multi-Objective Optimisation

We assume familiarity with genetic programming (GP) [Koz92,BNKF98,Lan98b]. In this work the GP uses Pareto tournament selection [Lan98b] with two objectives: maximize fitness and minimise size. We define program size as the number of functions and terminals it is made from. This has the advantage of simplicity and very compact solutions are evolved. However it does appear to squeeze out expressions involving constants. This may be disadvantageous, particularly where required a constant value is not included in the terminal set. Without fitness sharing the GP population tends to cluster around certain points on the Pareto front. In our experiments, without sharing, large fractions of the population converged to trees of one, two or three nodes. Fitness sharing is implemented by adding a second stage to tournaments which contain both larger higher fitness and shorter lower fitness individuals. Pareto comparison alone cannot chose between these. Instead such ties are resolved by comparing against the (a sample of) the rest of the population and preferring individuals from sparsely populated parts of the multi-objective (fitness-size) space. (Details in [Lan98b]). If a minimum solution size can be estimated, it might be worth including a size threshold, whereby tiny programs below the threshold are not preferred to bigger programs (but still below the threshold) of the same performance.

2 Insurance Customer Profiling

The task is given 85 attributes relating to a customer, such as age, number of children, number of cars, income, other insurance policies they hold, predict if they want caravan insurance. 5922 records (of which 343 are positive and the rest negative) are available as training data from `http://www.swi.psy.uva.nl/benelearn99/comppage.html` as part of the Benelearn'99 workshop. The task is to find 800 records of a further 4000 records which contain as many positive examples as possible.

We conducted two experiments. One with GP starting from a random population and the other where every member of the initial population was a copy of a seed individual created by C4.5 release 8 [Qui93]. (The unpruned C4.5 trees produced using default parameter settings were used).

The 5922 records were randomly split in half. One half (with 179 positive examples) was used as the training set and the other half was used as a verification set. Performance on it was reported by the GP but it was not used for selection during evolution. The details are given in Table 1, parameters not shown are as [Koz94, page 655].

Table 1. Insurance Customer Profiling

Objective:	Find a program that predicts the most likely $1/5^{th}$ of people to become caravan insurance customers.
Terminal set:	One terminal per data attribute, 0..41 and 110 different random numbers uniformly selected from 0..10 (total 255 primitives)
Function set:	IF IFLTE MUL ADD DIV SUB AND OR NAND NOR XOR EQ APPROX GTEQ LTEQ GT LT NOT
Fitness cases:	2911 (179 positive)
Hits:	number of positive cases predicted
Fitness:	At end of each generation each positive fitness case given a weight equal to the reciprocal of the number of individuals which correctly predicted it. Fitness given by sum of weights of positive cases predicted.
Selection:	Pareto tournament group size of 7 (fitness and size), non-elitist, generational. Fitness sharing (comparison set 81) [Lan98b].
Wrapper:	2911 values sorted, top 1/5 (583) treated as positive predicted
Pop Size:	100, 1000, 5000, 20000
Max prog. size:	no limit
Initial pop:	Created using "ramped half-and-half" with a minimum depth of 5 and a maximum depth of 9.
Parameters:	90% one child crossover, 5% point mutation (rate 10/1024), 5% size fair mutation (max subtree size 30) [Lan98a] 90% of crossover points selected at functions, remaining 10% selected uniformly between all nodes.
Termination:	Maximum number of generations G = 100

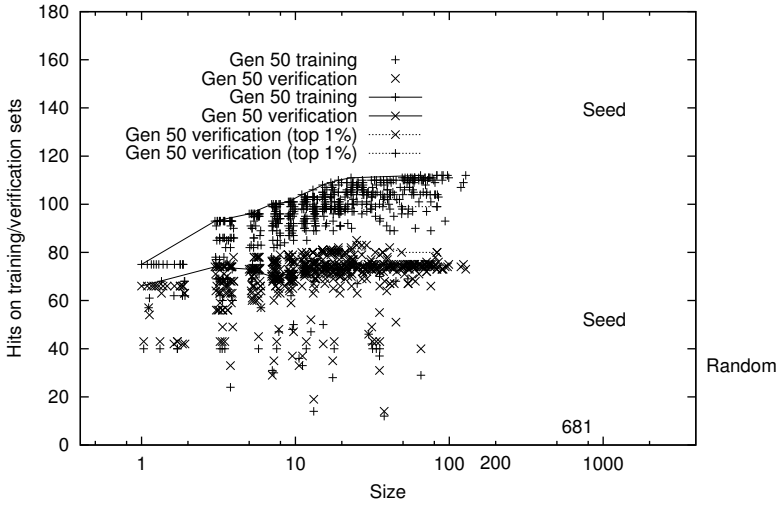

Fig. 1. Distribution of training (+) and verification (×) performance in the final generation of a non-seeded run of the customer profiling problem. Solid lines indicate individuals on the Pareto (training, size) front, while dotted lines indicate the best 1% of the population on the verification set. Note log scale.

Figure 2 shows the distribution of performance on both the training and verification sets in the final population. We see how the population has changed dramatically from the initial seed. As expect the initial seed performs reasonably well on the training set (from which was created) but only slightly better than random guessing on the verification set. By the end of the run the population is widely spread. Looking at performance on the training set, most of the population lies close to the Pareto front (solid line) (but some are markedly worse than it). Since GP has discovered programs that are shorter and/or fitter than the seed, the front is well to the left of it. After 100 generations the population has bloated as is indicated by the cluster of points at the top left. These programs are long and have high training scores but while longer they do not score markedly more than shorter programs in earlier generations. In fact while almost all long programs (e.g. longer than the seed 681) score better than it on the training case, they are worse than it on the verification set. Scarcely better than random guessing would achieve. That is, as might be expected, the population contains long programs which are heavily over trained.

The performance on the verification set of the best members on the training set is also plotted (lower solid line). Above size ≈ 200 many programs perform above the line on the verification set. I.e. many programs do better on the verification set than the best (on the training set) programs.

The lower dotted line shows the best 1% of the population on the verification set. This line is almost flat, Showing that bigger programs give little if any real performance advantage. Indeed in this population programs as short as five

appear to be the best. The upper dotted line shows the performance on the training set of the same 50 programs. Not surprisingly it is more erratic than the lower curve and climbs with programs size, again indicating over training.

It is clear that GP has been able to generalise the initial seed program and thereby considerably improve its performance. However there is also clear signs of over training and this increases with size of programs in the population.

The unseeded GP does not start from an over trained population. After a period of evolution the population shows only a little sign of over training. The generalisation performance appears approximately the same in the two cases.

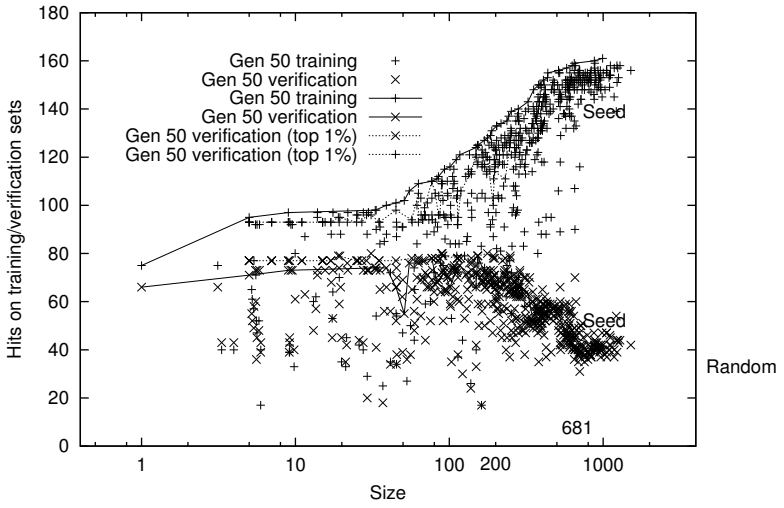

Fig. 2. Distribution of training (+) and verification (×) performance in the final generation of the first seeded run of the customer profiling problem. Solid lines indicate individuals on the Pareto (training, size) front, while dotted lines indicate the best 1% of the population on the verification set. Note log scale.

Both GP approaches suffered over fitting, that is, performance estimated from the verification set was not reached on the unseen data. Again high lighting the verification problem.

Acknowledgments

This research was funded by in part by the Wennergren foundation.

References

BNKF98. Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming – An Introduction; On the Automatic Evo-*

lution of Computer Programs and its Applications. Morgan Kaufmann, dpunkt.verlag, January 1998.

Koz92. John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

Koz94. John R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge Massachusetts, May 1994.

Lan98a. W. B. Langdon. The evolution of size in variable length representations. In *1998 IEEE International Conference on Evolutionary Computation*, pages 633–638, Anchorage, Alaska, USA, 5-9 May 1998. IEEE Press.

Lan98b. William B. Langdon. *Data Structures and Genetic Programming: Genetic Programming + Data Structures = Automatic Programming!* Kluwer, Boston, 24 April 1998.

Qui93. J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.