Sensitivity-Like Analysis for Feature Selection in Genetic Programming

Grant Dick
Department of Information Science
University of Otago
Dunedin, New Zealand
grant.dick@otago.ac.nz

ABSTRACT

Feature selection is an important process within machine learning problems. Through pressures imposed on models during evolution, genetic programming performs basic feature selection, and so analysis of the evolved models can provide some insights into the utility of input features. Previous work has tended towards a presence model of feature selection, where the frequency of a feature appearing within evolved models is a metric for its utility. In this paper, we identify some drawbacks with using this approach, and instead propose the integration of importance measures for feature selection that measure the influence of a feature within a model. Using sensitivity-like analysis methods inspired by importance measures used in random forest regression, we demonstrate that genetic programming introduces many features into evolved models that have little impact on a given model's behaviour, and this can mask the true importance of salient features. The paper concludes by exploring bloat control methods and adaptive terminal selection methods to influence the identification of useful features within the search performed by genetic programming, with results suggesting that a combination of adaptive terminal selection and bloat control may help to improve generalisation performance.

CCS CONCEPTS

•**Computing methodologies** → **Genetic programming; Feature selection; Model verification and validation;** *Classification and regression trees;* Supervised learning;

KEYWORDS

feature selection; variable importance, genetic programming, symbolic regression, random forests, CART

ACM Reference format:
Grant Dick. 2017. Sensitivity-Like Analysis for Feature Selection in Genetic Programming. In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071338

GECCO '17, Berlin, Germany

978-1-4503-4920-8/17/07...$15.00
DOI: http://dx.doi.org/10.1145/3071178.3071338

1 INTRODUCTION

Machine learning methods are being applied to increasingly large data sets, often with many input features. Given that the computational cost of machine learning typically increases in the presence of more features, and the quality of input features in large data sets often varies, it is often important to select a useful subset of features from the problem domain to produce accurate models in a timely manner [9]. It is for this reason that feature selection remains a widely explored concept in machine learning. Feature selection has also received considerable attention in evolutionary computation (EC) research, including genetic programming (GP). It is argued that feature selection is built directly into the symbolic regression process of GP, and that models evolved through symbolic regression can provide important insights into the useful features in the problem domain.

Typically, GP research has focused on a 'presence' interpretation of feature selection — if a feature regularly appears in the trees evolved by GP, then it must be useful. However, more research has questioned the usefulness of this approach on at least some problems [7]. This paper will compare the presence models used in GP with an alternative variable importance measure inspired by methods used in random forest methods. We show that the presence measures, even when considered over multiple runs, do not provide a clear measure of the true importance of a feature for a given problem. Solutions produced by GP often contain additional features that do not contribute greatly to the behaviour (and therefore fitness) of the model. The cause of this is often attributed to bloat, but this paper demonstrates that these extraneous features continue to appear in solutions even when bloat control measures are used. This suggests that such features are being used in a secondary manner, such as being used to evolve constant terms.

Typically, feature selection in GP is considered a by-product of the evolutionary process, and so within a run features are selected for use in tree generation and mutation with uniform probability. This paper explores the use of variable importance measures within a run of GP to adjust the selection probabilities of terminals. The resulting methods typically offer improved model performance over standard GP approaches, and the resulting feature distributions within the evolved models demonstrate less use of extraneous features, suggesting simpler models are being evolved.

The remainder of this paper is structured as follows: §2 discusses relevant work pertaining to feature selection and variable importance in machine learning and EC; §3 demonstrates the basic feature selection properties of GP on a range of synthetic and benchmark problems; §4 takes the findings of the previous section

and examines the use of bloat control and adaptive terminal selection to influence feature selection in GP; finally, §5 concludes the paper and suggests areas for future work.

2 FEATURE SELECTION METHODS

Feature selection in machine learning is generally considered to be the process of taking a larger set of input features and reducing the size of this set without compromising subsequent modelling performance. Feature selection is often used to remove noisy or uncorrelated variables from the data set prior to modelling, and is also used to provide insights to the user in terms of domain knowledge that may aid in decision making. Essentially, feature selection is a search and optimisation problem, where the optimisation typically makes a trade-off between feature set size and resulting modelling performance. Many approaches to feature selection have been proposed in the past, and essentially fall into three categories: *wrapper* approaches, where a given modelling process is embedded into a search algorithm to perform feature selection; *filter* methods, which do not use specific modelling techniques, but rather use information measures within the search process; and *embedded* techniques that integration feature selection directly into the learning algorithm. An excellent overview of these techniques is provided by [9], and the reader is directed to this work for more details. This paper focuses on embedded forms of feature selection.

As feature selection is essentially a search problem, it should be no surprise that it has been the subject of investigation in EC research. Much of this work has focused on genetic algorithms or particle swarm optimisation in a wrapper or filter approach: a comprehensive review of these techniques is provided in [19]. In the context of EC, this paper focuses on the embedded feature selection properties of genetic programming. In GP, it is argued that limits on tree size, coupled with selection pressure towards fitter solutions, places greater emphasis on features that will contribute positively towards fitness [1, 7, 13, 14, 17]. However, obtaining a useful feature selection process from GP often requires a filtering or multi-stage process where individuals must meet a certain criteria before they are accepted into the feature analysis process [4]. This added complexity might detract from making GP a useful feature selection algorithm.

Feature selection techniques are sometimes integrated directly into the process of machine learning methods. In these integrated forms the term feature selection is less common, and instead the notion of *variable importance* is more frequently encountered. For example, classification and regression tree (CART) methods acknowledge that merely being chosen as a split variable in a decision tree is not an adequate measure of the utility of a feature [3]. CART is a greedy algorithm, so a variable that is almost as useful as the chosen split variable will by masked, giving the false impression that this variable is not important. Therefore, the CART method uses the concept of surrogate splits to identify useful and potentially important variables that may have been masked by the splitting process. The resulting algorithm is then used to provide a relative ranking of the importance of a variable, usually normalised to a scale of $[0, 1]$, or $[0, 100]$, with larger values indicating greater variable importance. More details on the variable importance method used in CART can be found in Breiman et al. [3].

Random forests are related to CART in that their base learner is usually a variant of a CART decision tree [2]. The power of random forests comes from taking multiple bootstrap re-samples of the training data, applying the base learner to each sample, and then combining the resulting models into an ensemble (a form of bootstrap aggregation, or bagging). Unlike straightforward CART approaches, random forests do not employ surrogate splits to handle missing values, so they must rely on a different means of measuring variable importance. The key to variable importance in random forests lies in the use of out-of-sample statistics provided by the bagging process. After each base learner in the forest is built, the values in the instances that were out of its learning sample are shuffled one by one and the resulting change in prediction error is noted. Variables that, once shuffled, result in a large increase in prediction error are considered more important than those producing smaller changes in error. A more detailed description of variable importance measures in random forests can be found in Breiman's original paper [2]. Essentially, this process can be considered a form of sensitivity analysis that measures the influence that a given feature has on the outputs produced by the ensemble [16].

With appropriate consideration, variable importance measures can be adopted in a GP framework. Previous work used a variable importance measure to ascertain the importance of variables in the bioavailability data set frequent used in GP research [7]. It was argued that using a variable importance approach provided a stronger signal to feature importance than a straightforward feature presence approach. Subsequent work has questioned the utility of this variable importance approach in the context of classification, suggesting that a simple presence model of feature selection offers better discrimination to identify useful features [12]. This work continues the investigation into the utility of variable importance measures GP.

3 FEATURE SELECTION IN GP

The aim of this paper is to explore the embedded feature selection properties of GP. We wish to explore just the GP mechanism itself, and will not devise more sophisticated wrappers around GP to assist the feature selection. We will limit our analysis to a mixture of synthetic and benchmark regression problems — classification problems are left for exploration in future work. The problems explored are of relatively small size — our aim is to explore the feature selection properties of GP under tightly controlled conditions so that fine-grained examination of the results (e.g., looking at specific features) can be performed more easily. Scalability of EC methods for feature selection is a known issue [19], so future work should examine the methods discussed here on larger problems.

This paper uses a largely standard variant of GP [11], and the results of this model are compared to the results of modelling through CART and random forests. Parameters for the GP system are outlined in Table 1, and were obtained through empirical trials. The parameters were considered good enough for the results presented in this paper, but are not necessarily optimal — achieving optimal performance on problems is not an objective of this work, so no attempt has been made to optimise the performance of any method to any great extent.

Table 1: Parameters used for all experiments in this paper.

Parameter	Setting
Population size	200
Generations	1000
Initialisation	Ramped Half-and-Half
Min. initial depth	2
Max. initial depth	6
Mutation	Subtree (max. depth 4)
Crossover	Subtree swap
Max. offspring depth	17
Crossover prob.	0.3
Mutation prob.	0.7
Selection	Tournament (size: 3)
Elitism	1 (fittest from previous generation)
Function set	$+, -, \times, \div$, sin, cos, exp, log
Terminal set	Input features: $x_1, x_2, \ldots, x_p$, plus ephemeral random constants from a uniform distribution over [-5, 5]

For all experiments, we will use a fairly straightforward training and testing configuration. For each problem 10 runs of 10-fold cross validation are performed on the available data. Both the CART and random forests use their own internal cross-validation or bootstrapping processes on the training partition to tune the resulting models. For GP, we devise a simple validation strategy: the supplied training data is split into two equal sized partitions, one for training and one for validation. Within each generation, error measures based on the training data are used for selection and replacement purposes. At the end of each generation, each population member is evaluated on the validation set, and a copy of the individual with the best validation fitness is made. This validation model is retained until a later generation discovers an individual with lower error on the validation set. At the end of the run, the individual with the best encountered validation fitness over all generations is returned as the representative model.

For all methods, we record the root mean square error (RMSE) of the model returned at the end of evolution/learning on the corresponding test partition of the data. In addition, for GP we measure two aspects about the way that features are used by the representative model. The first is similar to the presence measurement from previous work [17] — the solution returned from evolution is analysed for presence/absence of a given feature, and then this process is repeated over multiple runs to obtain the mean presence of a given feature. As an extension of the presence measurement, we also measure the proportion of usage a feature received within the evolved models — the total number of feature terminals used in a given model is identified, and this is used to scale the number of times a given feature is contained within a given model. Both the presence and proportional feature selection statistics return a value between zero and one, with larger values indicating stronger usage of a given feature.

For all three methods, we also compute a variable importance statistic. For CART and random forest, this is the same process as defined in their respective works [2, 3]. For GP, the following process is used:

(1) At the end of each run, return the 'best' individual from the population (best is defined using the validation set process described above).
(2) Apply the chosen model to the hold-out partition of data and compute the model's mean squared error on the test data.
(3) For each variable, shuffle the values in the test data, reapply the model and compute mean squared error, making a note of the change in error from the unmodified test data. Once this is done, restore the variable's values back to their pre-shuffled state.
(4) Normalise the changes in mean squared error against the variable with the largest absolute change in error.

For each run, this process results in a value for each variable between ± 1, with a value of 1 indicating that the given variable had the most influence on the model. In some cases, a variable may receive negative influence (i.e., a score less than zero), which suggests that error was reduced by shuffling the variable — this is more likely to be observed for noise variables. At the end of the 100 runs, the influence measures for each variable are averaged to provide an indicator of its importance.

Table 2: Benchmark problems used in this paper.

Problem	#Inputs	#Instances	References
Auto MPG	7	392[1]	[5, 15]
Boston Housing	13	506	[3, 6, 10]
Concrete Strength	8	1030	[6, 20]
Servo	4	167	[2, 15]

Five synthetic data sets are used in this paper. Each data set consists of 1000 instances of 10 inputs variables sampled uniformly over $[1, 11]$. Then, for each problem, the response variable was established using each of the following function generators:

$$F_1(X) = x_1 + x_2 + x_3 + x_4 + x_5 \tag{1}$$

$$F_2(X) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 \tag{2}$$

$$F_3(X) = x_1 + \frac{x_2 x_3}{x_4} + \frac{x_3 x_4}{x_5} \tag{3}$$

$$F_4(X) = x_1 x_2 + x_2 x_3 + x_3 x_4 x_5 + x_5 x_6 \tag{4}$$

$$F_5(X) = \sqrt{x_1} + \sin(x_2) + \log_e(x_3) \tag{5}$$

Once the response variable is computed, random Gaussian noise is added to the variable to create a signal to noise ratio of 10. The resulting data sets, despite their relatively low dimensionality, should present a range of difficulties for GP, CART, and random forests, as they contain a number of interactions or variable transformations that must be learned before the problem can be solved. Because we know the underlying structure of each of the synthetic problems, we can perform some initial analyses of the relative importance of each feature. The importance measure technique previously defined for GP was used to compute the importance of each feature and the results of this are shown in Figure 1.

[1]The original data set is larger: instances with missing values were removed, as was the unique identifier column. This is normal practice for this data set.

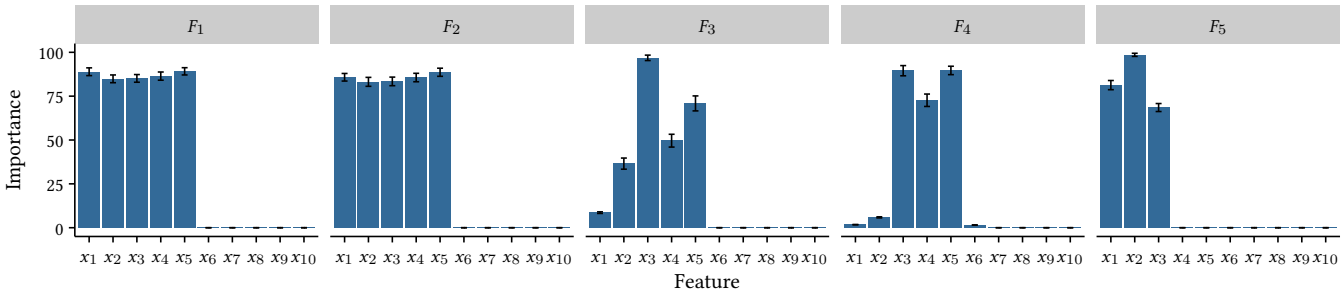

Figure 1: The underlying feature importance for each variable in the synthetic test problems.

In addition to the five synthetic problems, we explore four benchmark problems used in previous work. Details of these problems is shown in Table 2. Again, these problems are of low dimensionality, but most are reasonably well understood and should allow for good insights into the behaviour of embedded feature selection.

3.1 Results

RMSE values for the best of run models are presented in Figure 2. Although this is not the primary focus of the paper, it should be noted that the performance of GP relative to the other methods is particularly good on the synthetic problems. On the other benchmark problems GP generally remains competitive with at least one of the other methods in terms of median performance.

The relative measures to indicate feature selection performance are presented in Figure 3. Here, the raw importance measurements have been rescaled to a maximum of 100 to allow comparison across methods. We start by analysing the GP solutions returned at the end of the first generation — these are the lightly shaded rows of Figure 3. As these solutions have not been subjected to crossover or mutation, and only limited selection pressure to identify the fittest solution, we do not expect much signal to be present. The results in Figure 3 confirm this, with no meaningful signal detected in any of the synthetic problems. Some signal appears to be forming in the four benchmark data sets, particularly in the *Auto MPG* and *Servo* problems. The *Housing* data set presents an interesting observation: one of the features in the problem (x_{11}, pertaining to pupil-to-teacher ratios) appears to be selected more frequently than others, both in terms of actual presence and proportion of usage. However, when this variable is measured for its importance, it is considered very weak (the negative score suggests that shuffling this variable tends to reduce test error rather than increase it).

In contrast to the presence-based feature selection measurements, the variable importance measures on solutions from the initial population present in interesting picture. Despite limited selection pressure to pick the candidate solution, some patterns toward variable importance on the synthetic programs is already evident. While the features used in the denominators of F_3 appear to be poorly detected, as is the feature of F_5 that is wrapped in a sine function, the other features appear to be picked up as important even in the initial population. The variable importance measures on the benchmark problems are not as convincing, however, with only one variable from the *Auto MPG* data set demonstrating any level of influence — in this case, the variable x_5, which relates to vehicle acceleration performance.

Statistics for the feature selection and importance measurements at the end of run are represented by the darker shaded panels in Figure 3. The simple presence measurements for feature selection are not convincing, particularly for the benchmark data sets where there is almost no variation in selection of features. The performance on the synthetic problems is somewhat better, although there is still a large amount of presence of the noise variables in each case. To a large extent, the proportional feature selection at the end of the run presents a scaled version of the presence measurement, although there appears to be a somewhat stronger signal on the benchmark data sets.

Again in contrast to the presence-based measurements, the variable importance measures at the end of run for GP appear to show strong signals towards certain features. On the synthetic problems, there is a clear distinction between features that strongly influence the underlying variation in the problem signal and those that are merely noise and do not contribute. There is strong agreement with the shape of the plots for the synthetic problems relative to the trends shown in Figure 1. The variable importance measures on the benchmark data are also interesting: the negative influence of x_{11} in the *Housing* problem from the initial population solutions seems to have been reduced, while the acceleration variable in the *Auto MPG* problem no longer seems to carry any importance. For the *Concrete* and *Servo* problems, it appears that the variable importance measures is targeting the same features as in the original population, albeit with an amplified signal. The variable importance measures for CART and random forest are also shown in Figure 3. There is fairly large agreement between all three methods in terms of their ranking of the features, although a couple of points are worth further comment. In general, the CART variable importance measures attribute a small amount of importance to noise variables, and this is not present in either GP or random forest results. Also, both random forest and CART appear to attribute slightly greater importance to the fifth feature of problems F_1 and F_2, whereas GP-based feature importance considers them more equally. Random forests also appear to slightly undervalue the second feature of the F_5 problem when comparing the importance measure of this feature to the results for GP and the 'correct' signal shown in Figure 1. Finally, there appears to be some difference in the way that CART and random forests consider features on the benchmark problems, particularly on the *Auto MPG* and *Concrete* data sets. GP appears to have more difficultly than the other methods in attributing importance to features in the *Auto MPG* problem — the features in this problem are quite highly correlated, and it is possible than within

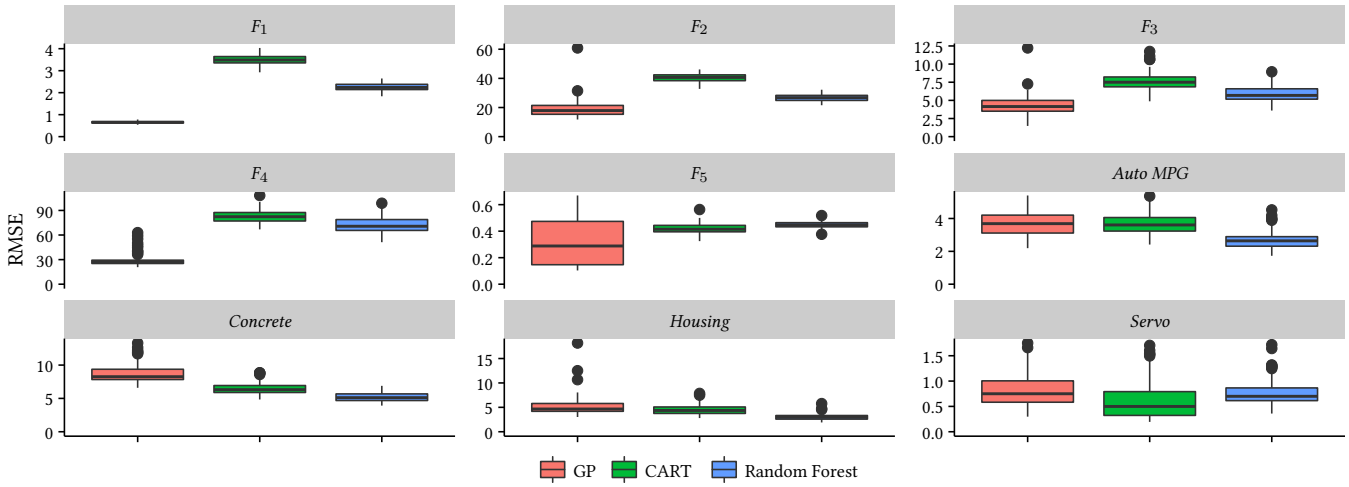

Figure 2: The test set error of the best model produced by each learning algorithm.

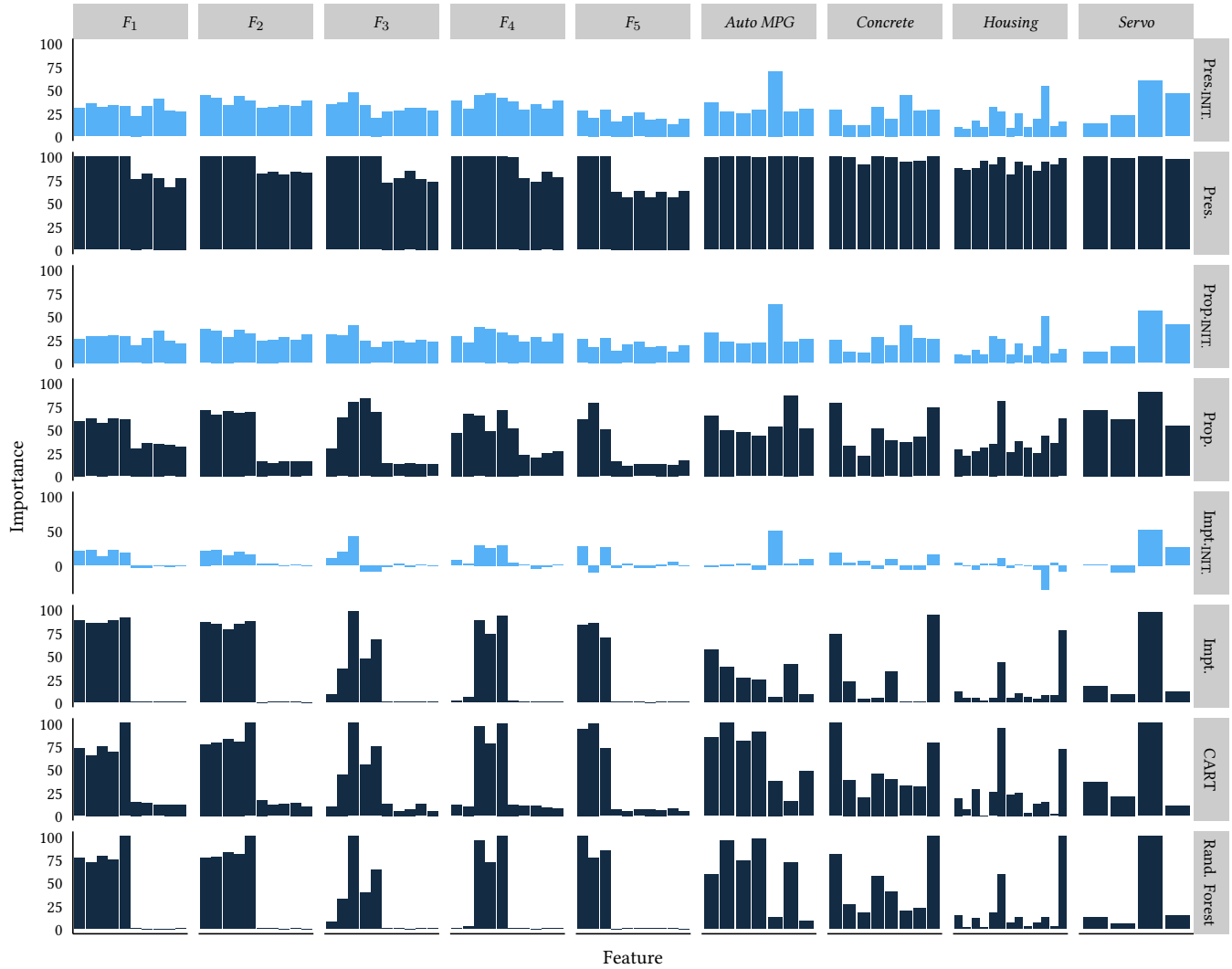

Figure 3: The measures of feature importance for each of the learning algorithms.

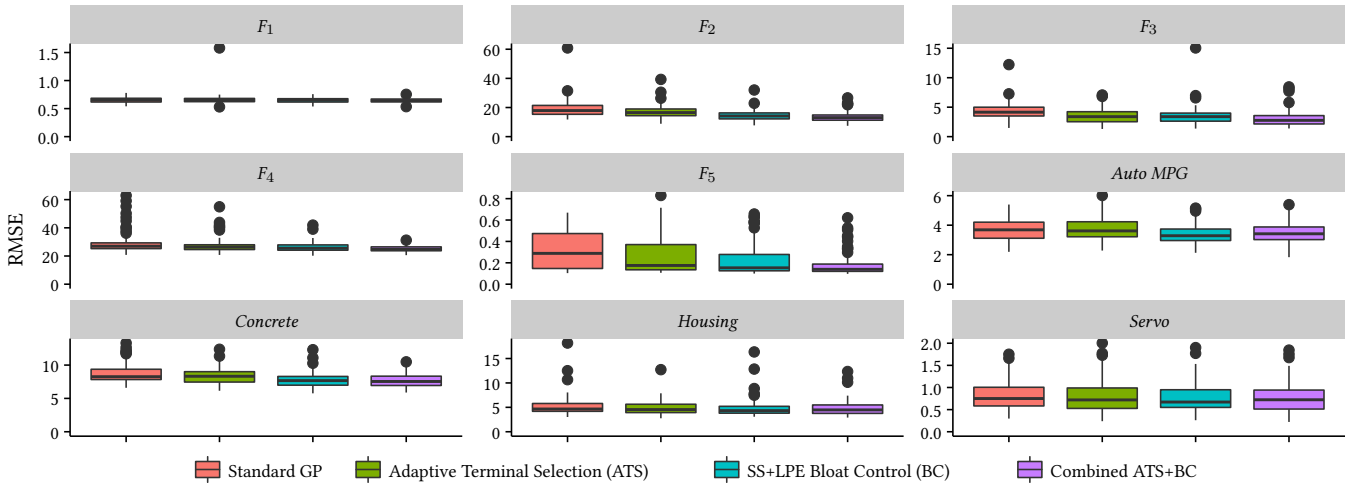

Figure 4: The test set error of best evolved models for each of the revised GP variants.

each run of GP, one particular feature is sampled more frequently than another and that this effect averages out over multiple runs.

Perhaps the most interesting result here is the strong difference between the presence-based feature selection and variable importance measures used in the GP runs. This is particularly evident in the benchmark data sets. The fact that almost all features appear in the best model in every run suggests that presence measures are not a good indicator of feature importance. Equally, the fact these features are present in models, but do not score highly in terms of variable importance measures, suggests two possible reasons for their inclusion in models:

(1) These features reside in areas of the tree that do not contribute to behaviour (i.e., they are present as a consequence of bloat). This might be a consequence of being present in a subtree that is ultimately paired with a multiplication by zero (for example).
(2) These features reside in actively executed regions of the tree, but are being used in ways that essentially nullify their variation (e.g., they are being used to create constants by being divided by themselves).

If either of these possible reasons is correct, then we should be able to identify solutions and modifications to GP that may help with stronger feature selection and usage.

4 EXTENDING FEATURE SELECTION WITHIN GP

The results in the previous section suggest that the internal mechanisms of standard GP do not provide meaningful feature selection. Once a feature is included in a tree, there does not seem to be much pressure for GP to eliminate unimportant features from the model. In this section, we examine two possible extensions to standard GP that may allow the evolutionary process to better exploit the underlying feature information.

One possible explanation for the excessive inclusion of features in evolved models may be through bloat — GP has a tendency to create very large trees where the majority of the tree does not contribute to the behaviour of the model. This creates a buffer for the evolved model and preserves the good properties already evolved, but limits opportunities for further evolving the behaviour of models. Previous work argues that bloat control may help with combating this phenomenon [8]. In particular, methods based around lexicographic parsimony, where the size of a tree is considered as a secondary selection factor in the presence of equal fitness, have been shown in previous work to be effective in reducing the amount of redundant code on models evolved by GP. In this section, we explore the use of one variant of lexicographic parsimony called SS+LPE (spatial structure with lexicographic parsimonious elitism) to attempt to control the unnecessary growth of individuals. Space precludes a full description of this method, so the readers are referred to previous work for details on how this approach works [5, 8, 18]. The expectation is that, if features are not being used in a meaningful way in models, then the bloat control provided by SS+LPE should work to remove them, resulting in simpler models with stronger focus on useful features. Conversely, if features are being used to produce constants (e.g., through division by self), then bloat control is not likely to alter feature selection.

In addition to exploring bloat control methods to hopefully improve feature selection, this section explores a simple mechanism to hopefully strengthen the internal feature selection properties of GP. While the results presented in the previous section for presence-based feature selection were not overly promising, the results of the variable importance analysis suggests that the evolved models exhibit good sensitivity towards variables known to be useful in a given problem. It would be interesting to see if this sensitivity information could be integrated into the evolutionary process of GP; doing so might improve the search properties of GP, or amplify the internal feature selection properties of GP. Stronger internal feature selection in GP may yield simpler solutions that are more interpretable. We therefore explore the following strategy to adapt the probability of selecting certain terminals at each generation:

(1) In the initial population, all terminal probabilities are equal.
(2) At the end of each generation, the solution with the best training fitness is identified and used to compute variable importance measures using the training data.

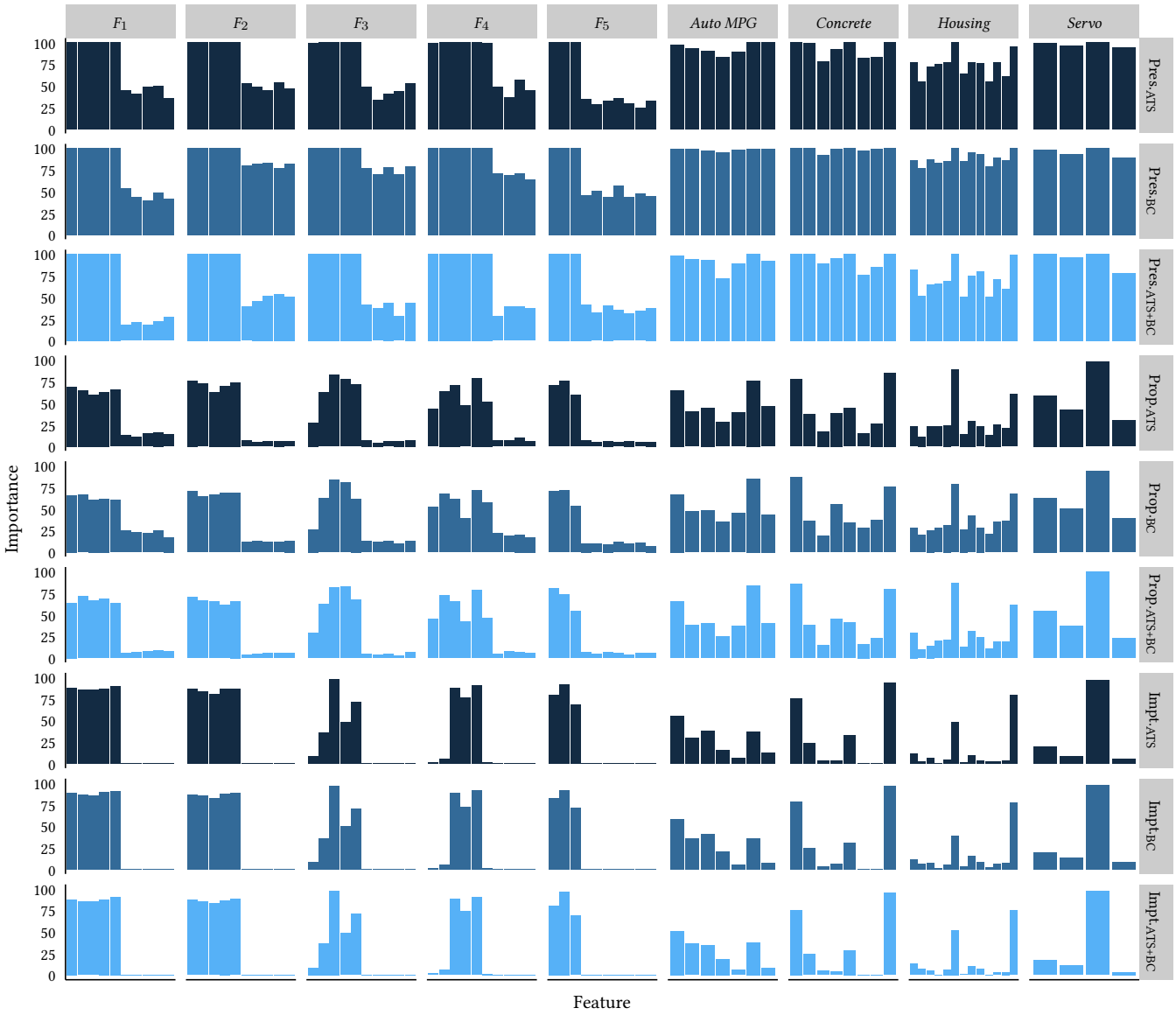

Figure 5: The measures of feature importance for each of the revised GP variants.

(3) The resulting variable importance measures are ranked (with ties given equal value).
(4) The ranks of the variables are then normalised by the sum of the ranks for all variables.
(5) In the next generation, the probability of choosing a given feature is proportional to its rank.

The intention of this adaptive process, which we refer to as *adaptive terminal selection*, is to increase selection pressure on the features that appear to be providing useful contributions to behaviour, and that this will lead to stronger selection pressure overall and improved model evolution.

In addition to exploring the bloat control (BC) and adaptive terminal selection (ATS) extensions separately, this section also explores a third variant that is the combination of the two methods (ATS+BC). The same nine problems were used as in the previous section, and the variants of GP were run using the same parameters outlined in Table 1. As before, 100 runs (10 runs of 10-fold cross validation) were used to determine average behaviour.

4.1 Results

The RMSE performance of the modified GP variants on the nine test problems is shown in Figure 4. Interestingly, in all cases there was at least some improvement in using the adaptive terminal selection approach. There was also some improvement in performance by using the SS+LPE bloat control method. Finally, the combination of both adaptive terminal selection and bloat control generally worked very well — notable improvements in fitness can be seen in F_5, *Auto MPG* and *Concrete* problems. As mentioned earlier, the

goal of this paper was not to find the best possible method, so it is possible that tuning the control parameters of GP may further improve performance.

A final comparison of the feature selection properties for the revised GP variants is shown in Figure 5. Surprisingly, none of the revised GP variants could substantially improve the straight presence-based measurement — there was some improvement on the synthetic problems when both bloat control and adaptive terminal selection were combined, but this improvement remained small. The variable importance measurement statistic already provided fairly strong signals, and the altered versions of GP did not substantially change this. The proportional-based measurement has changed slightly, and gave slight improvements that were more pronounced on the synthetic problems.

Given that the SS+LPE bloat control method was not able to substantially alter the composition of features in models, it seems likely that features are not entering into the population in a meaningless manner (i.e., they are not residing in parts of the tree that do not contribute to behaviour). More likely is that features are being exploited using 'self' operations to produce constants. This production of constants still needs to happen under the presence of adaptive terminal selection, however because the selection properties between features are no longer uniform, the choices of features is biased towards the more salient features, with the less useful features being less frequently selected. This might explain why the straight presence-based measurement was not greatly altered, while the proportional-based measurement showed a slight increase in emphasis towards stronger features.

5 CONCLUSION

The selection of useful features with which to conduct learning is vital aspect of all machine learning tasks. Genetic programming is frequently considered to provide embedded feature selection as part of the evolutionary process. This paper has explored the nature of feature selection in GP through a range of synthetic and benchmark problems. The results suggest that GP is indeed able to identify salient features for a given problem, but that caution must be used when adopting a simple measure of feature selection that considers only the presence of a feature in a model as an indicator of importance. Instead, the results from this paper suggest that a more useful feature importance measure is provided through a sensitivity-like analysis monitoring of changes in error rates, similar to that already used by other machine learning methods. Additionally, this paper has shown that this variable importance measure can be reintegrated into the inner workings of GP to bias feature selection when selecting terminals for tree construction. The early results presented here suggest that this approach may improve the quality of the solutions that are evolved with relatively little computational expense.

The work presented here raises a couple of areas of future work. Most importantly, this paper has only taken a quantitative approach to examining feature selection in GP. The next logical step is to examine the actual models that are evolved with and without adaptive terminal selection to see if there are any marked changes in the types of models that are evolved. Additionally, the variable importance measure used here only considers one variable at a time — higher level interactions between variables are not considered. Given the results presented here, where GP was able to easily identify and exploit the variable interactions in the synthetic problems F_3 and F_4, future work should focus on developing variable importance measures that consider feature interactions. Finally, the adaptive terminal selection method presented could be further developed, and future work in refining this approach may yield an even more effective system.

REFERENCES

[1] Francesco Archetti, Stefano Lanzeni, Enza Messina, and Leonardo Vanneschi. 2006. Genetic programming for human oral bioavailability of drugs. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 255–262.

[2] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[3] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. 1984. *Classification and regression trees*. CRC press.

[4] Q. Chen, B. Xue, B. Niu, and M. Zhang. 2016. Improving generalisation of genetic programming for high-dimensional symbolic regression with feature selection. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. 3793–3800. DOI: http://dx.doi.org/10.1109/CEC.2016.7744270

[5] Grant Dick. 2014. Bloat and generalisation in symbolic regression. In *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer International Publishing, 491–502.

[6] Grant Dick. 2015. Improving Geometric Semantic Genetic Programming with Safe Tree Initialisation. In *European Conference on Genetic Programming*. Springer International Publishing, 28–40.

[7] Grant Dick, Aysha P Rimoni, and Peter A Whigham. 2015. A re-examination of the use of genetic programming on the oral bioavailability problem. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 1015–1022.

[8] Grant Dick and Peter A Whigham. 2013. Controlling bloat through parsimonious elitist replacement and spatial structure. In *European Conference on Genetic Programming*. Springer Berlin Heidelberg, 13–24.

[9] Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of machine learning research* 3, Mar (2003), 1157–1182.

[10] David Harrison and Daniel L Rubinfeld. 1978. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management* 5, 1 (1978), 81–102.

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

[12] Róisín Loughran, Alexandros Agapitos, Ahmed Kattan, Anthony Brabazon, and Michael O'Neill. 2017. Feature selection for speaker verification using genetic programming. *Evolutionary Intelligence* (2017), 1–21.

[13] Durga Prasad Muni, Nikhil R Pal, and Jyotirmay Das. 2006. Genetic programming for simultaneous feature selection and classifier design. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 36, 1 (2006), 106–117.

[14] Kourosh Neshatian and Mengjie Zhang. 2009. Genetic programming for feature subset ranking in binary classification problems. In *European conference on genetic programming*. Springer, 121–132.

[15] J Ross Quinlan. 1993. Combining instance-based and model-based learning. In *Proceedings of the Tenth International Conference on Machine Learning*. 236–243.

[16] Andrea Saltelli, Karen Chan, E Marian Scott, and others. 2000. *Sensitivity analysis*. Vol. 1. Wiley New York.

[17] Sean Stijven, Wouter Minnebo, and Katya Vladislavleva. 2011. Separating the wheat from the chaff: on feature selection and feature importance in regression random forests and symbolic regression. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*. ACM, 623–630.

[18] Peter A Whigham and Grant Dick. 2010. Implicitly controlling bloat in genetic programming. *IEEE Transactions on Evolutionary Computation* 14, 2 (2010), 173–190.

[19] Bing Xue, Mengjie Zhang, Will N Browne, and Xin Yao. 2016. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation* 20, 4 (2016), 606–626.

[20] I-C Yeh. 1998. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete research* 28, 12 (1998), 1797–1808.