On the Effect of Function Set to the Generalisation of Symbolic Regression Models

Miguel Nicolau
University College Dublin
Dublin, Ireland
Miguel.Nicolau@ucd.ie

Alexandros Agapitos
EvoSys Analytics
Dublin, Ireland

ABSTRACT

Supervised learning by means of Genetic Programming aims at the evolutionary synthesis of a model that achieves a balance between *approximating* the target function on the training data and *generalising* on new data. In this study, we benchmark the approximation / generalisation of models evolved using different function set setups, across a range of symbolic regression problems. Results show that Koza's protected division and power should be avoided, and operators such as analytic quotient and sine should be used instead.

KEYWORDS

Symbolic Regression, Genetic Programming, Machine Learning, Generalisation, Overfitting, Data-driven Modelling

ACM Reference Format:
Miguel Nicolau and Alexandros Agapitos. 2018. On the Effect of Function Set to the Generalisation of Symbolic Regression Models. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, Article 4, 2 pages. https://doi.org/10.1145/3205651.3205773

1 INTRODUCTION

Supervised learning has two conflicting objectives, *approximating* a target function on the training data, and *generalising* on new examples not seen during training. Genetic Programming (GP) [4] searches a model space, populated by compositions of primitive functions that are collectively defined in a primitive *function set*.

Since the target function is unknown and GP searches for models in symbolic form, GP practitioners often resort to function sets containing diverse general-purpose functions, hoping that the resulting model space will contain a good model, and hoping that the training data will enable GP to pin down this model. The choice of primitive functions is largely dependent on function sets that are traditionally used in GP literature. A look at some of the most recently published GP applications to symbolic regression reveals that the vast majority of papers contain either the subset of basic arithmetic operators of addition, subtraction, multiplication, and some variant of protected division, or that subset combined with other popular operators, such as sine ($\sin(x)$) and cosine ($\cos(x)$), and protected versions of logarithm ($\log(x)$), square root ($\sqrt{x}$), and exponential functions.

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07.
https://doi.org/10.1145/3205651.3205773

Table 1: Function sets under comparison

Set ID	add	sub	mul	pdiv	AQ	plog	psqrt	ppow	sin	tanh
1	✓	✓	✓	✓	-	-	-	-	-	-
2	✓	✓	✓	-	-	-	-	✓	-	-
3	✓	✓	✓	-	-	✓	-	-	-	-
4	✓	✓	✓	-	-	-	✓	-	-	-
5	✓	✓	✓	-	✓	-	-	-	-	-
6	✓	✓	✓	✓	-	✓	✓	✓	✓	✓
7	✓	✓	✓	-	✓	-	✓	-	✓	✓
8	✓	✓	✓	-	-	-	✓	-	✓	✓
9	✓	✓	✓	-	-	-	-	-	-	-
10	✓	✓	✓	-	-	-	-	-	✓	-
11	✓	✓	✓	-	-	-	-	-	-	✓
12	✓	✓	✓	-	-	-	-	-	✓	✓

The current study compares different function set compositions, across a large set of symbolic regression problems. The results obtained highlight the detrimental effects that some operators have on the generalisation of GP-evolved models, and propose alternative function sets that are more robust, in terms of approximation (training error) and generalisation (test error) performance.

2 GP OPERATORS

Koza [4] defined *protected* versions of arithmetic functions, to avoid the generation of undefined values. These are:

- protected division: $pdiv(x_1, x_2) = x_1/x_2$ if $x_2 \neq 0$ else 1;
- protected logarithm: $plog(x) = log(|x|)$ if $x \neq 0$ else 0;
- protected square root: $psqrt(x) = \sqrt{|x|}$;
- protected power: $ppow(x_1, x_2) = |x_1|^{x_2}$ if $x_1 \neq 0$ else 0.

The problem with these functions is the presence of asymptotes within the function output values, which hinder generalisation performance. To address this, Keijzer [2] proposed the use of *interval arithmetic*, consisting in the static calculation of minimum and maximum bounds for each function of the function set, thus detecting and removing models leading to infinities and undefined values. Ni et al. [5] proposed instead to replace `pdiv` with Analytic Quotient (AQ), defined as: $AQ(x_1, x_2) = x_1/(\sqrt{1 + x_2^2})$.

3 EXPERIMENT DESIGN

To test the function sets defined, we used a large set of 42 recommended benchmark problems from recommended publications [2, 3, 7, 8] and real-world problems from the UCI ML Repository. We trained on 100 samples, and tested on 100,000 samples [5], using RMSE.

We used two GP systems, standard expression-tree GP and Grammatical Evolution (GE), with standard experimental setups [1, 6].

Table 1 specifies the function sets considered in this study.

To quantify relative performance for individual function sets, we show distributions of RMSE ratios. For each problem and function set, the mean RMSE (across 50 runs) on that problem is divided by the smallest mean RMSE obtained on that problem, over all function sets being compared. Function sets that are superior concentrate the mass of the distribution at the value 1. For visualisation purposes, we cap the RMSE ratios to the value of 5.

4 RESULTS

4.1 Approximation performance

We first benchmark approximation performance (RMSE on training data) of the different function sets; this is plotted in Figure 1 (top), for GP. Large function sets 6, 7, 8 enable the best training performance, with regard to the RMSE Ratio. Of interest is the remarkably good training performance attained using the smaller function set 10. This suggests that GP is capable of approximating arbitrary functions provided that the function set contains a Tauber-Wiener function such as sine and a linear function.

The worst performing set (set 9) is the one composed solely of the three arithmetic operators. Although it should be noted that sets 1-4 exhibit inconsistency in the results obtained for different problems. Finally, comparing sets 1 and 5 shows that replacing Koza's protected division with AQ seems to improve the approximation capabilities of the evolved models.

4.2 Generalisation Performance

Generalisation performance is plotted in Figure 1 (bottom). There is a decidedly different performance between sets. Sets 1, 2 and 6, containing the protected division and/or the protected exponentiation operators, consistently generate mean test RMSE performance which is substantially worse than all other sets.

Sets 3 and 4 show that the protected logarithm (3) and protected square-root (4) operators do not seem to generate such extreme values, with the latter providing marginally better test performance (for both GP and GE).

Comparing sets 1 and 5 shows again that replacing `pdiv` with `aq` results in models with better test performance. Function sets 7, 8, 10 and 12 consistently produced a good approximation / generalisation trade-off. There exists a well-performing interaction between the operators of AQ, sine, sqrt, and hyperbolic tangent, which is performance-analogous to the interaction of sine, sqrt, and hyperbolic tangent alone. The results of set 6 demonstrate that good approximation performance of large function sets containing protected operators leads to bad generalisation.

5 CONCLUSION

Koza's protected operators of division and power resulted in the evolution of models that do not generalise, and should be avoided. Protected operators of logarithm and square root are virtually pathology-free and can be used in large diverse functions sets to improve approximation performance. Analytic Quotient remains an excellent alternative to division. Furthermore, the combination of sine, addition, subtraction and multiplication has interesting theoretical properties, and was empirically shown to yield good relative performance on both training and testing datasets, which is similar to that of best-performing AQ setups.

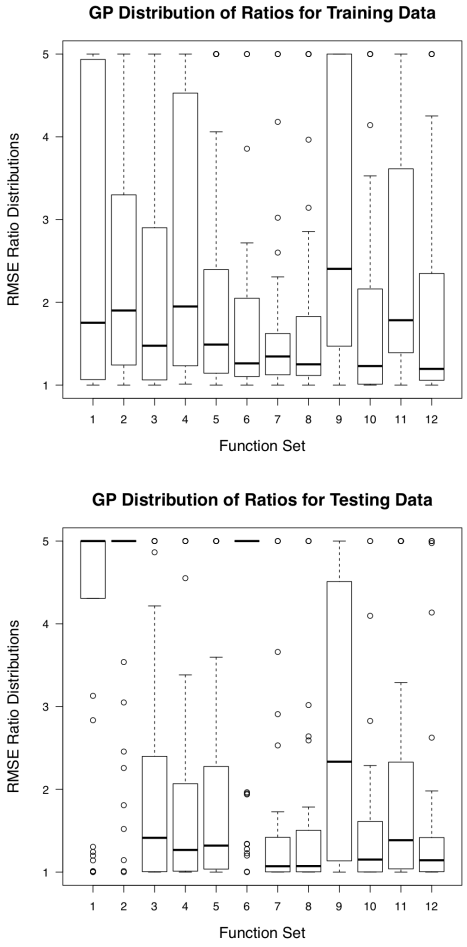

Figure 1: Mean RMSE ratio distributions for all function sets, over 42 problems, for train (top) and test (bottom), using GP (GE results were similar). Results obtained across 50 independent runs for each function set / problem combination.

REFERENCES

[1] Alexandros Agapitos, Julian Togelius, and Simon Mark Lucas. 2007. Evolving controllers for simulated car racing using object oriented genetic programming. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, Dirk Thierens et al. (Ed.), Vol. 2. ACM Press, London, 1543–1550.

[2] Maarten Keijzer. 2003. Improving Symbolic Regression with Interval Arithmetic and Linear Scaling. In *Genetic Programming, Proceedings of EuroGP'2003 (LNCS)*, Conor Ryan et al. (Ed.), Vol. 2610. Springer-Verlag, Essex, 70–82.

[3] Michael F. Korns. 2011. Accuracy in Symbolic Regression. In *Genetic Programming Theory and Practice IX*, Rick Riolo et al. (Ed.). Springer, New York, 129–151.

[4] J.R. Koza. (1992). *Genetic Programming: on the programming of computers by means of natural selection.* MIT Press, Cambridge, MA.

[5] Ji Ni, Russ H. Drieberg, and Peter I. Rockett. 2013. The Use of an Analytic Quotient Operator in Genetic Programming. *IEEE Transactions on Evolutionary Computation* 17, 1 (Feb. 2013), 146–152.

[6] Miguel Nicolau. 2017. Understanding grammatical evolution: initialisation. *Genetic Programming and Evolvable Machines* 18, 4 (Dec 2017), 1–41.

[7] Ludo Pagie and Paulien Hogeweg. 1997. Evolutionary Consequences of Coevolving Targets. *Evolutionary Computation* 5, 4 (1997), 401–418.

[8] Ekaterina J. Vladislavleva, Guido F. Smits, and Dick den Hertog. 2009. Order of Nonlinearity as a Complexity Measure for Models Generated by Symbolic Regression via Pareto Genetic Programming. *IEEE Transactions on Evolutionary Computation* 13, 2 (April 2009), 333–349.