Empirical Analysis of Variance for Genetic Programming based Symbolic Regression

Lukas Kammerer
Josef Ressel Center for Symbolic Regression
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Campus Hagenberg
Johannes Kepler University
Department of Computer Science
Linz, Austria
lukas.kammerer@fh-hagenberg.at

Gabriel Kronberger
Josef Ressel Center for Symbolic Regression
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Campus Hagenberg
Hagenberg, Austria
gabriel.kronberger@fh-hagenberg.at

Stephan Winkler
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Campus Hagenberg
Johannes Kepler University
Department of Computer Science
Linz, Austria
stephan.winkler@fh-hagenberg.at

ABSTRACT

Genetic programming (GP) based symbolic regression is a stochastic, high-variance algorithm. Its sensitivity to changes in training data is a drawback for practical applications.

In this work, we analyze empirically the variance of GP models on the PennML benchmarks. We measure the spread of model predictions when models are trained on slightly perturbed data. We compare the spread of models from two GP variants as well as linear, polynomial and random forest regression models.

The results show that the spread of models from GP with local optimization is significantly higher than that of all other algorithms. As a side effect of our analysis, we provide evidence that the PennML benchmark contains two groups of instances (Friedman and real-world problem instances) for which GP performs significantly different.

CCS CONCEPTS

• **Computing methodologies** → *Supervised learning by regression*; Optimization algorithms; • **Software and its engineering** → **Genetic programming**;

KEYWORDS

Genetic Programming, Symbolic Regression, Bias/Variance Tradeoff

ACM Reference Format:
Lukas Kammerer, Gabriel Kronberger, and Stephan Winkler. 2021. Empirical Analysis of Variance for Genetic Programming based Symbolic Regression . In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459486

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459486

1 INTRODUCTION

Symbolic regression is a machine learning task wherein we search for a mathematical human-readable expression that covers dependencies within data. Especially the human readable representation is a crucial feature in many applications. However, each GP run will yield a different model due to its stochasticity. This puts practitioners in the dilemma, that they have to run GP multiple times but might end up with several models that seem equally suitable.

In this work we analyze how much the predictions of GP models from different GP runs differ from each other. We analyze, if GP finds models that produce completely dissimilar outputs for the same inputs, or if the found models differ just in their structure but provide similar prediction behavior. We introduce the *phenotypical spread* as a new measure for variance of an algorithm. We apply it on regression models from GP, GP with nonlinear least squares optimization (GP NLS) [3], linear regression (LR), polynomial regression (PR) and random forest regression [1]. The models are trained on the problems of the the PennML benchmark library [4] – a large set of machine learning benchmark problem instances.

2 PHENOTYPICAL SPREAD

The phenotypical spread describes how far the predictions of a set of models spread on average on a given dataset. We define the phenotypical spread $s(\boldsymbol{m}, D)$ as follows: Consider a dataset $D \in \mathbb{R}^{n \times d}$ of n observations of d features and a set $\boldsymbol{m} = \{m_1 ... m_k\}$ of k models. Each model m_j in $\boldsymbol{m}$ is applied on every observation in D to create a set $\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \{m_1(D_i) ... m_k(D_i)\}$ of m predictions for each observation D_i. We calculate for every observation, how far these m predictions in $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ spread using the interquartile range $\mathrm{IQR}(\hat{\boldsymbol{y}}_{\boldsymbol{i}})$. The phenotypical spread $s(\boldsymbol{m}, D)$ is the average of those n interquartile ranges.

$$s(\boldsymbol{m}, D) = \frac{1}{n} \sum_{i=1}^{n} \mathrm{IQR}\left(\left\{m_1(D_i) ... m_k(D_i)\right\}\right) \tag{1}$$

We generate for each problem instance a new dataset D with 1000 samples and calculate $s(\boldsymbol{m}, D)$ for each algorithm. To generate one sample D_i in D, we select for each feature a random entry of that feature from the original dataset. Our measure $s(\boldsymbol{m}, \boldsymbol{x})$ is similar to the "error due to variance" in the bias/variance decomposition of

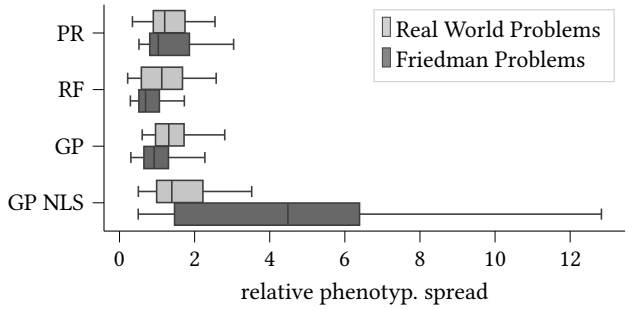

Figure 1: Distribution of relative phenotypical spread. All values are relative to the values of linear regression models.

the mean squared error in [2]. The major difference is the use of the interquartile range, which is robust against prediction outliers in extrapolation. Such outliers can occur because we do not preserve dependencies between features during the generation of D, which leads to extrapolation when the models $\boldsymbol{m}$ are applied.

For each problem instance and algorithm, we tune the algorithms' hyperparameters with a grid search with a 5-fold cross-validation on the original dataset. We then train 50 models $\boldsymbol{m} = \{m_1 ... m_{50}\}$ with the best hyperparameter setting. We randomly choose 75% of the dataset's observations for the training of each model the use the remaining 25% as test partition, so that every model is trained on slightly perturbed data. Both GP flavors use the same hyperparameter configuration as in [3]. The parameters of the non-evolutionary algorithms, which are optimized in the grid search are the following:

- RF with 200 trees, $m \in \{0.25, 0.5, 0.75\}$, $r \in \{0.1, 0.2, \ldots, 0.7\}$
- PR is implemented as elastic-net regression (cf. [5]) with polynomial basis function expansion. We tested $\alpha \in \{0, 0.5, 1\}$, $\lambda \in \{1 \cdot 10^{-7}, 2.5 \cdot 10^{-7}, 5 \cdot 10^{-7}, 7.5 \cdot 10^{-7}, 1 \cdot 10^{-6}, 2.5 \cdot 10^{-6} \ldots 7.5 \cdot 10^{-2}\}$ and total polynomial degree $\in \{2, 3, 4, 5\}$. For instances with > 20 features we limited the degree to 3.

3 RESULTS

Our experiments show that there are clear differences between two sets of problems in the PennML benchmark suite regarding both phenotypical spread and prediction accuracy. Figure 1 shows the distribution of the relative phenotypical spread values for all algorithms and problem instances. Since the value range of the phenotypical spread is problem-specific, we relativize the values of all algorithms to the one of linear regression, so they are on the same scale. Values for LR are left out as they are always one.

Figure 1 shows that the phenotypical spread of GP NLS models is much higher than the spread of models from other algorithms in the Friedman problems. GP and RF provide similar spread in the Friedman problems, while PR exhibits slightly higher spread. In contrast, all algorithms perform similar in the real world problems. Although GP NLS also has the highest spread among real world problems, the difference to the other algorithms are less pronounced.

The real world problems also differ from the Friedman problems concerning prediction error. Figure 3 outlines how all algorithms provide similar prediction error on the real world problems. LR is only slightly worse and RF over-fits but still performs competitive on test. On the Friedman problems, only GP NLS performs well both in test and training as shown in Figure 2. The other algorithms achieve only low Pearson's R^2 values in test there.

Given the clear differences in the modelling results between the Friedman problems and the real world problems, as well as the larger number of Friedman problems, algorithm tests on the whole PennML benchmark library are biased towards the Friedman problems. Algorithms that excel on the Friedman problems will also excel when analyzed on all problems. We suggest for future work to use these two groups of problems separately.

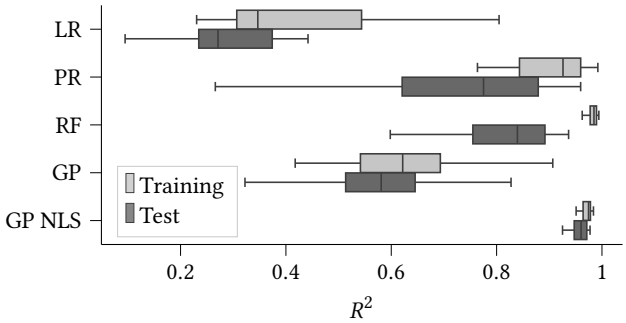

Figure 2: Median R^2 for Friedman problems.

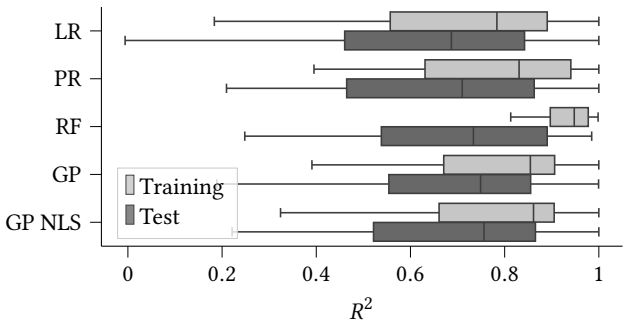

Figure 3: Median R^2 for real world problems.

ACKNOWLEDGMENTS

The authors gratefully acknowledge support by the Christian Doppler Research Association and the Federal Ministry of Digital and Economic Affairs within the *Josef Ressel Center for Symbolic Regression*

REFERENCES

[1] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[2] Maarten Keijzer and Vladan Babovic. 2000. Genetic Programming, Ensemble Methods and the Bias/Variance Tradeoff – Introductory Investigations. In *European Conference on Genetic Programming*. Springer, 76–90.
[3] Michael Kommenda, Bogdan Burlacu, Gabriel Kronberger, and Michael Affenzeller. 2019. Parameter identification for symbolic regression using nonlinear least squares. *Genetic Programming and Evolvable Machines* (2019), 1–31.
[4] Randal S. Olson, William La Cava, Patryk Orzechowski, Ryan J. Urbanowicz, and Jason H. Moore. 2017. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining* 10 (2017), 36.
[5] Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B* 67, 2 (2005), 301–320.