Bias-Variance Decomposition: An Effective Tool to Improve Generalization of Genetic Programming-based Evolutionary Feature Construction for Regression

Hengzhe Zhang, Qi Chen, Bing Xue, Wolfgang Banzhaf, Mengjie Zhang

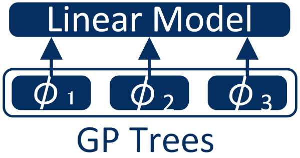
Victoria University of Wellington

15/07/2024

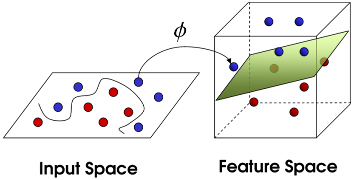
Table of Contents

Background

- **Objective:** Construct a set of new features, $\{\phi_1, \ldots, \phi_m\}$, to *enhance learning* on the dataset $\{\{x_1, y_1\}, \ldots, \{x_n, y_n\}\}$ compared to learning on the original features $\{x^1, \ldots, x^p\}$.
- **Approaches:**
  - Kernel Methods: Black-box, non-parametric.
  - Neural Networks: Black-box, gradient-based.
  - Genetic Programming: Interpretable, gradient-free.



(a) Feature Construction on Linear Regression



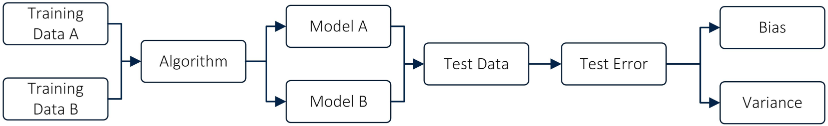
(b) New Feature Space

- **Challenge:** Overfitting is a significant issue in evolutionary feature construction.
- **Phenomenon:** Overfitted models perform well on training data but poorly on unseen data.
- **Cause:** Models may become too complex and fit noise in the training data, especially when:
  - Sample size is limited.
  - Data contains noise.

Objective

Mitigate overfitting to enhance generalization and robustness of constructed features.

- **Concept:** Bias-variance decomposition separates prediction error into:
  - **Bias:** Error from incorrect assumptions in the learning algorithm (e.g., linear regression).
  - **Variance:** Error from sensitivity to small changes in the training data (e.g., very large neural network).
  - **Irreducible error:** Error due to noise in the data.
- **Objective:** Reduce variance to improve generalization.



Bias-Variance Decomposition Workflow

Method

- **Framework:** Use bias-variance decomposition to reduce variance and improve generalization.
- **Decomposition:**

$$\mathbb{E}_{\mathcal{D}}\left[\{f(\mathbf{x}_{\mathbf{test}};\mathcal{D})-y_{test}\}^2\right]=\underbrace{\{\mathbb{E}_{\mathcal{D}}[f(\mathbf{x}_{\mathbf{test}};\mathcal{D})]-y_{test}\}^2}_{\text{Bias}} \tag{1}$$

$$+\underbrace{\mathbb{E}_{\mathcal{D}}\left[\{f(\mathbf{x}_{\mathbf{test}};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[f(\mathbf{x}_{\mathbf{test}};\mathcal{D})]\}^2\right]}_{\text{Variance}} \tag{2}$$

- **Bias:** The squared difference between the expected prediction and the actual target value.
- **Variance:** The expected squared deviation of the prediction from its expected value.
- **Goal:** Optimize both bias and variance to achieve better generalization.

- **Approach:** Estimate variance by adding noise to the training data.
- **Steps:**
  1. Add Gaussian noise to the training data to get $x + \epsilon$.
  2. Construct features $\Phi(x + \epsilon)$.
  3. Make predictions using the constructed features to get $LM(\Phi(x + \epsilon))$.
  4. Measure the variance of predictions $(LM(\Phi(x + \epsilon)) - LM(\Phi(x)))^2$.
- **Objective Functions:**

$$O_1(\Phi) = \frac{1}{|X|} \sum_{x \in X} (LM(\Phi(x)) - Y)^2 \tag{3}$$

$$O_2(\Phi) = \frac{1}{|X|} \sum_{x \in X} (LM(\Phi(x + \epsilon)) - LM(\Phi(x)))^2 \tag{4}$$

Algorithm Framework Summary

- **Task:** Multi-tree GP for feature construction on a linear regression model.
- **Objectives:** Minimize **cross-validation loss** and **estimated variance**.
- **Process:**
  - **Population Initialization:** Initialize population with GP trees.
  - **Fitness Evaluation:** Evaluate individuals using cross-validation loss and the proposed variance estimation method.
  - **Parent Selection:** Select parents using **lexicase selection**.
  - **Offspring Generation:** Generate offspring with GP operators, including random tree addition/deletion and random subtree crossover/mutation.
  - **Environmental Selection:** Use **NSGA-II** for environmental selection.
- **Final Model:** Selected from the Pareto front based on the sum of cross-validation loss and estimated variance.

Experimental Settings

- **Source:** 42 real-world datasets from the Penn Machine Learning Benchmark (PMLB).
- **Criteria:** Excluded synthetic datasets and those with fewer than 5 features.
- **Focus:** Emphasis on real-world applicability.

Parameter Settings for GP

Parameter	Value
Maximal Population Size	200
Number of Generations	200
Crossover and Mutation Rates	0.9 and 0.1
Tree Addition Rate	0.5
Tree Deletion Rate	0.5
Initial Tree Depth	0-3
Maximum Tree Depth	10
Initial Number of Trees	1
Maximum Number of Trees	20
Elitism (Number of Individuals)	1

Parameter	Value
Standard Deviation of Noise	0.5
Iterations of Variance Estimation	5
Functions	+, -, *, AQ, Square, Log, Sqrt, Max, Min, Sin, Cos, Abs, Negative

Compared Methods:

- Standard GP without regularization
- Parsimonious Pressure (PP)
- Tikhonov Regularization (TK)
- Grand Complexity (GC)
- Rademacher Complexity (RC)
- Weighted MIC between Residuals and Variables (WCRV)
- Correlation between Input and Output Distances (IODC)

Results

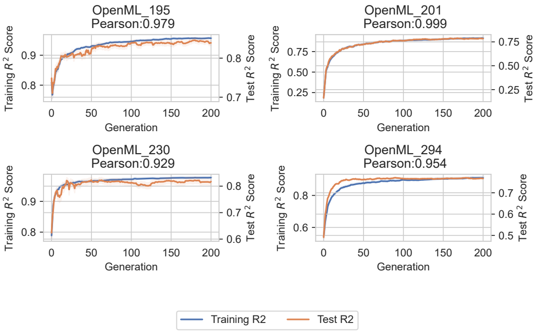
- **Objective:** Compare test R^2 scores across methods.
- **Outcome:** VR method significantly improves generalization.

Statistical comparison of test R^2 scores.

	PP	RC	GC	IODC	TK	WCRV	Standard GP
VR	23(+)/10(∼)/9(-)	32(+)/10(∼)/0(-)	23(+)/15(∼)/4(-)	31(+)/10(∼)/1(-)	35(+)/5(∼)/2(-)	22(+)/14(∼)/6(-)	35(+)/2(∼)/5(-)
PP	—	24(+)/13(∼)/5(-)	17(+)/17(∼)/8(-)	18(+)/18(∼)/6(-)	18(+)/24(∼)/0(-)	17(+)/18(∼)/7(-)	28(+)/10(∼)/4(-)
RC	—	—	6(+)/11(∼)/25(-)	11(+)/15(∼)/16(-)	9(+)/13(∼)/20(-)	5(+)/12(∼)/25(-)	17(+)/5(∼)/20(-)
GC	—	—	—	22(+)/17(∼)/3(-)	20(+)/20(∼)/2(-)	14(+)/23(∼)/5(-)	20(+)/15(∼)/7(-)
IODC	—	—	—	—	11(+)/20(∼)/11(-)	12(+)/14(∼)/16(-)	19(+)/10(∼)/13(-)
TK	—	—	—	—	—	6(+)/23(∼)/13(-)	16(+)/16(∼)/10(-)
WCRV	—	—	—	—	—	—	17(+)/15(∼)/10(-)

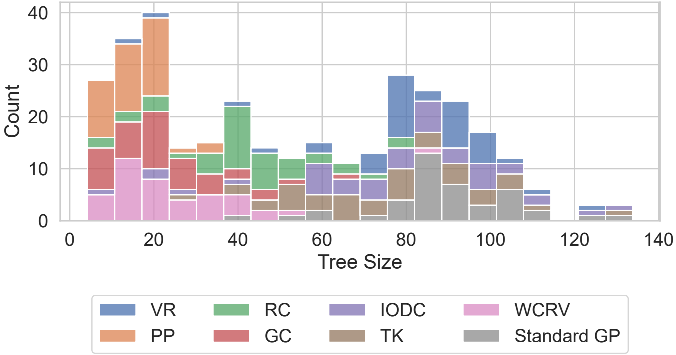
Training Performance

- **Objective:** Calculate correlation between training and test R^2 scores.
- **Outcome:** VR method effectively controls overfitting.



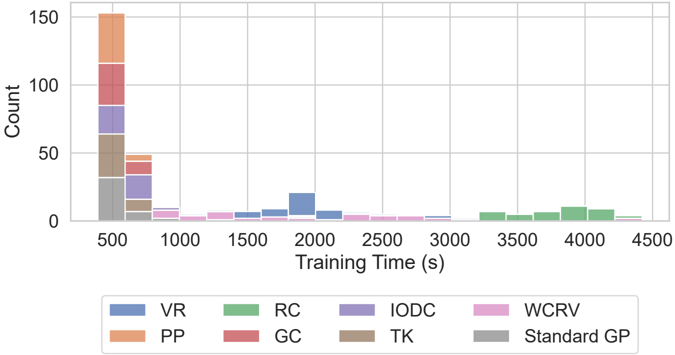
Evolutionary plots of the *training and test R^2 scores* for VR.

- **Objective:** Compare tree sizes across methods.
- **Outcome:** VR does not significantly reduce tree size compared to standard GP.



Distribution of tree sizes across methods.

- **Objective:** Compare training times across methods.
- **Outcome:** VR method is computationally more intensive.



Distribution of training time across methods.

Conclusions

Key Takeaways

- Optimizing variance based on bias-variance decomposition improves generalization.
- VR outperforms standard GP and other overfitting control techniques.

Thanks for listening!

Email: Hengzhe.zhang@ecs.vuw.ac.nz

GitHub Project: https://github.com/hengzhe-zhang/EvolutionaryForest