Discovering Test Statistics Using Genetic Programming

Jason H. Moore*
Institute for Biomedical Informatics
University of Pennsylvania, Philadelphia, PA
jhmoore@upenn.edu

Randal S. Olson
Institute for Biomedical Informatics
University of Pennsylvania, Philadelphia, PA
rso@randalolson.com

Yong Chen
Institute for Biomedical Informatics
University of Pennsylvania, Philadelphia, PA
ychen123@pennmedicine.upenn.edu

Moshe Sipper
Institute for Biomedical Informatics
University of Pennsylvania, Philadelphia, PA
& Dept. of Computer Science, Ben-Gurion University
Beer-Sheva, Israel
sipper@upenn.edu

ABSTRACT

We describe a genetic programming-based system for the automated discovery of new test statistics. Specifically, our system was able to discover test statistics as powerful as the t-test for comparing sample means from two distributions with equal variances [1].

CCS CONCEPTS

• **Mathematics of computing** → **Probability and statistics**; **Evolutionary algorithms**; • **Theory of computation** → **Evolutionary algorithms**;

KEYWORDS

Genetic Programming, Statistics, Optimization, T-Test

ACM Reference format:
Jason H. Moore, Randal S. Olson, Yong Chen, and Moshe Sipper. 2019. Discovering Test Statistics Using Genetic Programming. In *Proceedings of GECCO '19 Companion, Prague, Czech Republic, July 13–17, 2019,* 2 pages.
https://doi.org/10.1145/3319619.3326754

The goal of the present study was to develop an evolutionary system for the automated discovery of new test statistics. There were three important challenges that needed to be addressed to accomplish this objective. First, we needed an engine for generating mathematical candidates for test statistics, in our case using available array-based operators in a modern programming language with a data structure that is easy for the computer to manipulate. Second, we needed a set of evaluation criteria that are general enough to allow the computer to generate innovative solutions while specific enough to satisfy human statistical objectives without directing the computer to pre-determined outcomes. Third, we needed a system that could tinker with candidate test statistics as a mathematician would, by making small changes and by interchanging functional modules to create new solutions. We applied a genetic programming (GP) solution to this problem.

In our GP experiments, we asked whether the GP system is capable of discovering test statistics similar in power to the t-test (Equation 1) when presented with our general evaluation criteria.

$$t = \frac{\overline{x_1} - \overline{x_2}}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}} \quad (1)$$

To answer this question, we first simulated data drawn from two normal distributions, each with different means (0 and 1, 0 and 2, or 0 and 4) but the same variances (standard deviations of 1, 2, or 4, respectively). A total of 30 data sets with sample size of n=100 were simulated for each set of means. These data are used for the evaluation criteria to represent data consistent with the alternate hypothesis of a difference in means. Next, we permuted each of these 30 data sets to create pairs of distributions consistent with the null hypothesis that the data were drawn from the same distribution with equal means and variances. Finally, we simulated data under the alternate hypothesis such that as the difference in the means increased (0 and 10 or 0 and 100) the equal variances also increased (10 or 100, respectively). A total of 30 datasets were simulated for each set of means and variances. These data were used to evaluate scale invariance. Finally, we implemented an evaluation criterion that encourages smaller models, to incentivize the GP system to explore smaller models while at the same time being able to create diversity by considering larger models.

The key to implementing an evolutionary approach to the discovery of test statistics is to articulate the objective criteria that are important to human statisticians. We chose here to focus on four very general criteria to allow the system to be innovative. First, we want a test statistic to have a low rate of type I errors. These occur when the null hypothesis is true but is rejected. Second, we want a statistic to have good power under the alternative hypothesis (power is the probability that the test correctly rejects the null hypothesis when a specific alternative hypothesis is true). Third, we want a statistic to be invariant to the scale of the data, making it generalizable. Finally, we would like a statistic to be as simple as possible, thus making it easy to understand and implement.

*J. H. Moore and R. S. Olson contributed equally to this paper.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326754

To quantify the objective criteria we used the following fitness function. When we evaluate an individual, we provide the aforementioned pairs of distributions to the evolved test statistic to generate the test statistic scores for the pairs. Next, we compute a Gaussian kernel density estimate (KDE) of the test statistics from the null distribution pairs (the null distribution is the probability distribution of the test statistic when the null hypothesis is true). The KDE allows us to measure the probability of a test statistic value appearing in the null distribution.

For the first objective—low false positive rate—we take the evolved test statistic values computed from the null distributions across each set of 30 data sets and measure the probability of them occurring in the null distribution. In this case, higher probabilities are considered better because it entails that the null test statistic values are distributed together around a single value.

For the second objective—high power—we take the evolved test statistic values computed from the distribution pairs with differences in means and measure the probability of them occurring in the null distribution. In this case, lower probabilities are considered better because it entails that the test statistic values, when there is a difference in means, fall outside the null distribution. We note that we combined objectives one and two into a single fitness component to limit the multiobjective search space, as the objectives are highly related.

For the third objective—scale invariance—we used the evolved test statistic values from the distribution pairs with means of 0 and 1, 0 and 10, and 0 and 100. As these distribution pairs are the exact same but with a multiplier of 1, 10, and 100, respectively, a test statistic that is scale invariant should produce the exact same test statistic values for these distributions. Thus, for this objective we considered lower sums of differences between the test statistic pairs to be better.

For the fourth objective—simplicity—we used the number of primitives in the GP tree as the measure of complexity. The number of primitives in the GP tree directly correlates to the complexity of the function; thus, GP trees with fewer primitives are considered better.

Using the GP system with these evaluation data and criteria, we ran 30 unique replicate runs (i.e., with different random seeds) with a population size of 1000 candidate test statistics for 1000 GP generations. We saved the entire Pareto front of test statistics at the end of every replicate run and manually inspected every test statistic on the final Pareto fronts.

Figure 1 shows the t-test as a GP tree and Figure 2 shows evolved solutions.

Across all 30 replicate runs, the GP system discovered test statistics *that had a fitness that was equal to or better than the t-test.*

Our results showed that in each of the replicate runs the GP system was able to generate test statistics that had fitness values as good as or better than the t-test that is the widely accepted and applied solution to this problem. Further, our GP-generated test statistics were linearly related to the t-test and tended to be much simpler. We concluded that GP is suited to the automatic generation of test statistics and should be extended and applied to unsolved test statistic problems in statistics.

The need for new statistics is exploding as new technologies give us new data with unique characteristics that yield new scientific

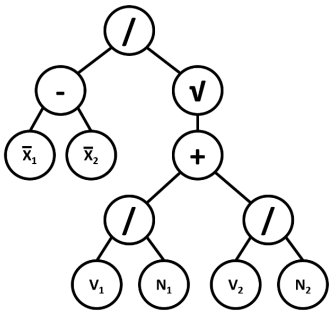

Figure 1: The two-sample t-test equation represented as a binary expression tree. The vector of sample values is represented by X, sample means by X bar, variances by V, and sample sizes by N for samples one and two.

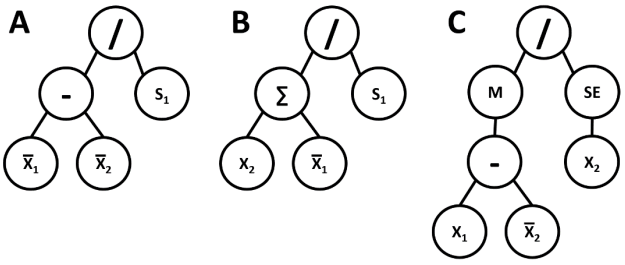

Figure 2: Three GP-generated test statistics represented as binary expression trees. The vector of sample values is represented by X, sample means by X bar, standard deviations by S, standard error by SE, and median by M.

questions. There is no question that data and experimental designs are changing at a rate that exceeds that of mathematical statisticians. In fact, the number of statisticians that actively develop new test statistics is decreasing as trainees opt for more exciting and lucrative fields such as data science, where the demand for the application of statistical methods and machine learning is exploding. This is the right time to explore artificial intelligence methods for assisting statisticians with the automated development of test statistics.

ACKNOWLEDGMENTS

The work is supported by the National Institutes of Health (USA) under Grants Nos.: LM012601, AI116794, DK112217.

REFERENCES

[1] Jason H. Moore, Randal S. Olson, Yong Chen, and Moshe Sipper. 2019. Automated discovery of test statistics using genetic programming. *Genetic Programming and Evolvable Machines* 20, 1 (2019), 127–137.