Logic Guided Genetic Algorithms (Student Abstract)

Dhananjay Ashok[1], **Joseph Scott**[2], **Sebastian J. Wetzel**[3], **Maysum Panju**[2], **and Vijay Ganesh**[2]
[1]University of Toronto, Canada
[2]University of Waterloo, Canada
[3]Perimeter Institute for Theoretical Physics, Canada
dhananjay.ashok@mail.utoronto.ca,{joseph.scott, mhpanju, vijay.ganesh}@uwaterloo.ca, swetzel@perimeterinstitute.ca

Abstract

We present a novel Auxiliary Truth enhanced Genetic Algorithm (GA) that uses logical or mathematical constraints as a means of data augmentation as well as to compute loss with the aim of increasing both data efficiency and accuracy of symbolic regression (SR) algorithms. Our method, logic-guided genetic algorithm (LGGA), takes as input a set of labelled datapoints and *auxiliary truths* (AT) (mathematical facts known a priori about the unknown function the regressor aims to learn) and outputs a specially generated and curated dataset that can be used with any SR method. We evaluate LGGA against state-of-the-art SR tools, namely, Eureqa and TuringBot, and find that using these SR tools in conjunction with LGGA results in them solving up to 30.0% more equations, needing only a fraction of the amount of data compared to the same tool without LGGA, i.e., resulting in up to a 61.9% improvement in data efficiency.

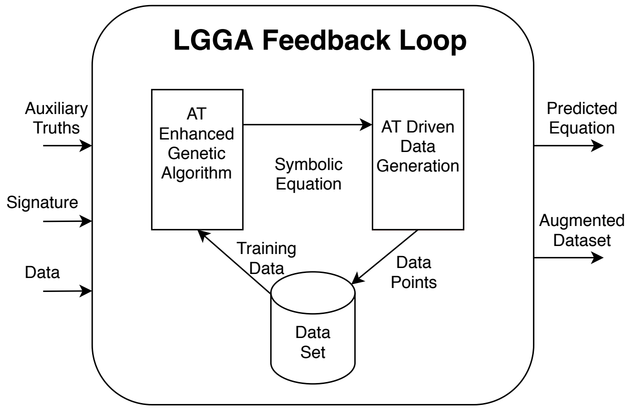

Figure 1: Architecture Diagram for LGGA

Introduction

The problem we aim to solve is the following: How can the user of a symbolic regression (SR) system leverage their domain-specific mathematical knowledge, i.e., auxiliary truths (ATs), with the aim of making SR more data efficient and more likely to find the target equation? This work is inspired by (Scott, Panju, and Ganesh 2020).

SR systems take as input a dataset and a set of symbols (input alphabet), and output a candidate symbolic expression over the input alphabet that fits the data. In LGGA, we also take the set of ATs. We define ATs as mathematical expressions that capture domain-specific knowledge or simple properties of an unknown function f to be learnt. For example, consider the equation $R = \frac{r_1 r_2}{r_1 + r_2}$ (Parallel Resistance). Physicists could draw upon their domain knowledge about resistors and infer an AT: if one resistor has zero resistance, the combined resistance is zero, a simple AT that a physicist could infer without knowing the actual equation. This AT can be exploited as an added specification to aid the SR procedure such that it is more data efficient and produces an expression that is a better approximation of the target equation/function than otherwise.

A crucial property of any AT we consider is the following: a candidate learnt function or symbolic expression is incorrect if it is inconsistent with the input ATs. Further, whenever an SR system produces a candidate symbolic expression, if such an expression happens to be inconsistent with the ATs (i.e., a counterexample can be computed), not only do we **receive a signal that the candidate function is incorrect** which can be used as part of a loss function, but we also can **use the counterexample to augment the dataset** and feed it back to the SR system in a corrective feedback loop.

AT Enhanced Loss Function: The AT Enhanced Loss Function is a weighted sum of the traditional Mean Squared Error (MSE) and *TruthError* – a measure which is higher for equations that have a higher degree of violation of the input ATs. To do this, we use a violation function $v_t(\widehat{f}, \mathbf{x})$ which is a measure of the violation of AT t for a given datapoint $\mathbf{x}$ when using a candidate equation $\widehat{f}$. If $v_t(\widehat{f}, \mathbf{x}) > v_t(\widehat{g}, \mathbf{x})$ it implies that $\widehat{f}$ violates the AT more on a given datapoint than another candidate equation $\widehat{g}$ does. We define Truth Error due to a single candidate function $\widehat{f}$ produced by an SR system as

$$\mathrm{TruthError}(\widehat{f}, T, \mathbf{X}) := \frac{1}{|T|} \sum_{t \in T} \max_{\mathbf{x} \in \mathbf{X}} (v_t(\widehat{f}, \mathbf{x})) \qquad (1)$$

where T is the set of all ATs known a priori of the target function f, $\mathbf{X}$ is dataset and v_t is the violation function for truth $t \in T$.

Equations	Minimum Datapoints Needed for Eureqa			Minimum Datapoints Needed for TuringBot		
Target Equation	LGGA	No LGGA	DE %	LGGA	No LGGA	DE %
$\frac{r_1 r_2}{r_1+r_2}$	8 ± 2	21 ± 3	62	1 ± 0	1 ± 0	0
$\frac{\sin(i)}{\sin(r)}$	6 ± 1	14 ± 2	58	6 ± 1	10 ± 2	40
$\frac{Kq_1q_2}{r^2}$	6 ± 1	14 ± 2	58	2 ± 0	2 ± 0	0
$\lvert\frac{n_1-n_2}{n_1+n_2}\rvert^2$	200 ± 40	**NoDisc**	**Disc**	600 ± 100	**NoDisc**	**Disc**
$\frac{PV}{nT}$	6 ± 3	15 ± 4	60	4 ± 0	4 ± 0	0
$\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}$	200 ± 100	**NoDisc**	**Disc**	2000 ± 800	**NoDisc**	**Disc**
$\frac{e^{-x^2}}{2\pi}$	190 ± 100	**NoDisc**	**Disc**	300 ± 50	800 ± 200	62.5
$\frac{Gm_1m_2}{\frac{1}{r_2}-\frac{1}{r_1}}$	13 ± 5	23 ± 3	45	10 ± 5	25 ± 7	60
$\frac{m_1r_1+m_2r_2}{m_1+m_2}$	22 ± 3	38 ± 5	43	7 ± 0	14 ± 3	50
$mrv\sin(\theta)$	23 ± 6	48 ± 7	52	11 ± 3	20 ± 5	45
$\sqrt{\frac{pr\gamma}{\rho}}$	23 ± 5	55 ± 6	59	1300 ± 750	2000 ± 700	35
$\frac{gqB}{2m}$	15 ± 4	40 ± 3	62.5	18 ± 3	25 ± 4	28

Table 1: Select results from Experiment. Abbreviations Used:- NoDisc: Does not Discover Equation. Disc: Enables Discovery.

AT Driven Data Augmentation: Unlike classic GAs, instead of having a fixed input dataset, the LGGA system progressively generates and adds *interesting* datapoints as the training goes on. Every time a new generation is created, we find the best performing function and use the points in the current dataset to check whether any AT is violated for this equation. If an AT is violated, we can use it to produce new datapoints from the existing ones without having to query an oracle to obtain output labels. This continues until the specified generation limit is met, or the overall error reaches a threshold. As an example, suppose we wish to learn an unknown function f where the given AT is that f is symmetric in its arguments, and are performing the data augmentation process for candidate equation $\widehat{f}(x_1, x_2)$. If the provided dataset contains the datapoint $f(2,3) = 4$, then to evaluate whether $\widehat{f}$ is compliant with the AT we would check if $|\widehat{f}(2,3) - \widehat{f}(3,2)| > 0$. At the same time, we are able to create a *new datapoint for free* which is $f(3,2) = 4$.
Encoding Boundary Points: The way a function behaves around key points of interest, including domain boundaries, strongly defines the function's behavior in general. ATs frequently target these key points when used for data augmentation, thus, augmenting a dataset with these points significantly reduces the set of viable choices of expressions that an SR system has to make.
Deterring Overfitting: Modern SR techniques avoid overfitting by penalizing high complexity expressions, allowing them only when they admit a low MSE. This is less effective with small datasets since overfitted equations can achieve a near-zero MSE when data is scarce. LGGA solves this by consider the AT enhanced loss function which penalizes AT violating functions even if they are good fits by MSE.

Experiments and Results

We test LGGA's ability to generate richer and more informative datasets to augment industrial-strength SR tools and quantify how effective this data augmentation procedure is in improving the highly-engineered SR tools Eureqa and TuringBot (Schmidt and Lipson 2009). We first find the minimum number of datapoints needed for Eureqa (resp. TuringBot) to solve the equation with a timeout of 15 minutes - m_{RAND}. We start with a large number of random datapoints and lower the number of points given until the tool stops solving the equation. We then produce an augmented dataset and follow the same procedure to find the minimum number of datapoints needed for Eureqa (resp. TuringBot) to solve the equation using the augmented dataset - m_{LGGA}. We then use as a metric the improvement in data efficiency: $\frac{m_{\mathrm{RAND}} - m_{\mathrm{LGGA}}}{m_{\mathrm{RAND}}}\%$ as a way to measure the reduction % of data needed, in an apple-to-apple comparison of Eureqa vs. Data Augmented Eureqa (resp. TuringBot). The above experiments were repeated five times for each equation, and we noted the mean and deviation of the minimum number of datapoints needed to learn each equation.

Eureqa shows significant improvements when used with an LGGA augmented dataset (shown in Table 1). Every equation shows a reduction in the minimum number of datapoints needed for discovery, and three equations are discovered only when LGGA augmented data is used. Even with these industrial-strength tools, we see a consistent reduction in the number of datapoints needed as well as enabling the discovery of two equations (see Table 1). This experiment shows that LGGA has a significant positive influence on the data efficiency and convergence rates of industrial SR tools. For more results see:https://dhananjayashok.github.io/LGGA/

References

Schmidt, M.; and Lipson, H. 2009. Distilling free-form natural laws from experimental data. *science* 324(5923): 81–85.

Scott, J.; Panju, M.; and Ganesh, V. 2020. LGML: Logic Guided Machine Learning (Student Abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, 10, 13909–13910.