Probabilistic Machine Learning Project

Zak Bastiani

Spring 2024

Introduction

Symbolic Regression (SR) is focused on finding the underlying function used to generate a dataset. SR has large applications within physics where knowing the equation is more important than having a black box predictor. Within the SR, some methods use reinforcement learning. These RL methods can be broken into two sections: genetic programming and deep learning. Genetic programming became the staple of SR around the turn of the century; however, in recent years, deep learning-based approaches have become more common in both a supervised and reinforcement learning framework. The most prominent deep learning model came from [1], where they introduced Deep Symbolic Regression (DSR).

DSR uses a RNN as the actor to generate a set of equations. This set of equations is typically called the population, τ, (because of GP), but in RL, it is also known as the environment state. Once all equations are generated, DSR uses a policy to select the best performers from the state to inform the actor of its performance for its actions. DSR chose a risk-seeking policy that only informs the actor of its top $\alpha\%$ of performers from the population. This risk-seeking policy causes the actor to take more risk when selecting equations as bad performers will not impact the actor's overall performance. Note that an additional self-entropy gradient was added to encourage the actor to explore the space, but is not the focus of this research.

Motivation

Reinforcement-based approaches to SR have seen a downturn in interest over the last few years, even though SR has become more popular. One of the primary reasons for this is that transformers can absorb and remember a large amount of data, allowing for an approach similar to LLMs to be taken [2]. However, DSR still performs close to the Pareto front for R^2 and model size. It shows how SR methods can select for accuracy at the cost of simplicity or vice versa. This research will focus on improving the risk-seeking policy within DSR. The risk-seeking policy is not prevalent in most RL applications; its counterpart, a risk-averse policy, has some research to improve it. The goal of this research is twofold: First, to test the effectiveness of improvements in risk-adverse policy to the risk-seeking policy. Second, see if an equation exists to update the risk-seeking policy for optimal performance dynamically.

Mathematical Backing

$$\nabla_\theta J_{\text{risk}}(\theta;\alpha) \approx \frac{1}{\alpha N}\sum_{i=1}^{N}[R(\tau^{(i)}) - \tilde{R}_\alpha(\theta)]\cdot \mathbf{1}_{R(\tau^{(i)})>\tilde{R}_\alpha(\theta)}\nabla_\theta \log p(\tau^{(i)}|\theta) \tag{1}$$

In DSR, they define a risk-seeking policy as given in equation 1 where R denotes the reward function, θ is the actor's weights, and α is the top x percent of the population [1]. This equation is a basic implementation of a risk-seeking gradient that gains the benefit of freely exploring the space while also exploiting the top x percent of the population. However, this implementation of a risk-seeking policy has a similar problem of *blindness to success* when the actor learns from prior non-optimal successes, which causes it to limit its effective exploration. If an actor is continuously affected by *blindness to success*, they will eventually encounter a tail barrier.

Tail Barrier: Let $\alpha \in (0, 1]$. A policy π has an α-tail barrier if $\forall \alpha' \in [\alpha, 1] : R(\tau^{(\alpha')}) = R(\tau^{(\alpha)})$ [3].

A tail barrier denotes the end of learning for the risk-seeking policy as the gradient calculated by equation 1 will go to 0. If the actor starts to suffer from blindness to success, they will be unable to find the optimal solution. [3] found an upper bound probability for this tail barrier to occur. Let m donate the training step, $\beta \in (0, \alpha)$, and α is the desired risk-seeking percentile.

$$A = \{\{\tau_{m,i}\}_{i=1}^{N} \in \mathcal{T}^N | q_\beta^{\pi_m} > \hat{q}_\alpha(\{R(\tau_{m,i})\}_{i=1}^{N})\} \tag{2}$$

$$p(A) \leq e^{-\frac{N(\beta-\alpha)^2}{2\beta(1-\beta)}} \leq e^{-2N(\beta-\alpha)^2} \tag{3}$$

A denotes the event where an equation that could have a high enough reward to be with the top $\alpha\%$ given another action is overlooked. If the event A occurs frequently, the actor is increasingly likely to encounter a tail barrier. The equation shows that the best way to prevent a tail barrier is to decrease the α. Thus, there is a crucial balancing act between exploration and exploitation, in which lower values of α allow for better exploration, while higher allow for better exploitation. With the risk-averse paper, they used an α that decreased linearly, which showed a substantial improvement in identifying the optimal path in a path-finding environment. Thus, I mirror the idea for a risk-seeking policy but increase it linearly, as shown in equation 4. In this case, the gradient calculated for each step in equation 1 starts with the top 30% of the population and linearly approaches α throughout the N training steps.

$$\alpha_i = \frac{i * (\alpha_0 - 0.3)}{N} + 0.3 \tag{4}$$

The second change I wanted to test was how removing the reward multiplier from the gradient would affect the performance, as shown in equation **??**. The reason behind removing the reward multiplier is twofold. First, the reward multiplier is more likely to cause success to blindness.

$$\nabla J_{\text{risk},i} = [R(\tau^{(i)}) - \tilde{R}_\alpha(\theta)] \cdot \mathbf{1}_{R(\tau^{(i)}) > \tilde{R}_\alpha(\theta)} \nabla_\theta \log p(\tau^{(i)}|\theta) \tag{5}$$

$$\forall R(\tau^{(i)}) \approx \tilde{R}_\alpha(\theta), \;\; \nabla J_{\text{risk},i} = 0 \tag{6}$$

Suppose we assume that the sampled population is normally distributed. In that case, it is fair to assume that a nontrivial portion of the top $\alpha\%$ of the population will have $R(\tau^{(i)}) \approx \tilde{R}_\alpha(\theta)$. Thus, the actor only learns from a much smaller subset of the top $\alpha\%$ of the population. I will later call this policy the "linear policy"

Lastly, lets look at the reward function. The reward function is just a non-linear mapping between the NMSE and a bounded dimension between $(0, 1]$. This non-linear mapping can cause tail-barriers to appear in epochs where the NMSE between two equations is significant.

Lemma: Any static function $f : x \to y$ where $x \in [0, \infty)$ and $y \in [0, 1]$ can create a tail barrier.

Proof: Assume we are given an arbitrary set $X \in [0, \infty)$. We must prove that for any constant function that maps $[0, \infty)$ there will exist a cluster of points where all $D(x_i, x_j) > \epsilon_{NMSE}$ where ϵ_{NMSE} is a numerically significant distance away, but $D(f(x_i), f(x_j)) < \epsilon_f$.

Assume there exists a function f_1 such that $D(f(x_i), f(x_j)) > \epsilon_f$ and $D(x_i, x_j) > \epsilon_{NMSE}$, then f must be linear function as the distance measure ϵ_{NMSE} and ϵ_f will be linear. However a linear function can not map between $[0, \infty)$ and $(0, 1]$ thus a contradiction.

Therefore, for any $f : NMSE \to (0, 1]$ a new tail barrier can exist.

Thus, we must select a dynamic function $f : NMSE \to (0, 1]$, however this can cause bias that is not in NMSE. Therefor, we propose selecting a dynamic function that minimizes its influence on the gradients. We want to minimize its influence on the gradients because, from lemma 1, the mapping of NMSE will cause information loss.

$$\begin{cases} D_\theta = 1 & z > Q_{1-\alpha}(Z; \theta) \\ D_\theta = 0 & z \leq Q_{1-\alpha}(Z; \theta) \end{cases} \tag{7}$$

Instead, we propose using a different uninformative function that is peicewise based on the $\alpha\%$. This dynamic function prevents any tail barrier from occurring as $\forall \alpha' \in (\alpha, 1) : q^\pi_{\alpha'} - q^\pi_\alpha = 1$

Building an unbiased risk seeking gradient

Consider an unbound random variable $Z \in [0, \infty)$, where Z is generate from a parameterized distribution $p(Z|\theta)$.

$$Q_{1-\alpha}(Z; \theta) = \inf\{z : \text{CDF}(z) \geq 1 - \epsilon\} \tag{8}$$

where $\text{CDF}(z)$ is the cumulative distribution of the parameterized distribution $p(Z|\theta)$.

$$\begin{cases} D_\theta = 1 & z > Q_{1-\alpha}(Z; \theta) \\ D_\theta = 0 & z \leq Q_{1-\alpha}(Z; \theta) \end{cases} \tag{9}$$

$$\int_{z \in D_\theta} p(z|\theta) dz = \alpha \tag{10}$$

Next, we can set up the risk seeking gradient as a probability density.

$$J_{risk}(\theta; \alpha) = \frac{1}{\int_{z \in D_\theta} p(z|\theta) dz} \int_{z \in D_\theta} p(z|\theta) z dz \tag{11}$$

$$= \frac{1}{\alpha} \int_{z \in D_\theta} p(z|\theta) z dz \tag{12}$$

$$= \frac{1}{\alpha} \int_{Q_{1-\alpha}(Z;\theta)}^{b} p(z|\theta) z dz \tag{13}$$

Here we are applying Leibniz rule, and simplifying

$$\nabla_\theta J_{risk}(\theta; \alpha) = \nabla_\theta \frac{1}{\alpha} \int_{Q_{1-\alpha}(Z;\theta)}^{b} p(z|\theta) z dz \tag{14}$$

$$= \frac{1}{\alpha} \int_{Q_{1-\alpha}(Z;\theta)}^{b} \nabla_\theta p(z|\theta) z dz - \frac{1}{\alpha} p(Q_{1-\alpha}(Z; \theta)|\theta) \nabla_\theta Q_{1-\alpha}(Z; \theta) \tag{15}$$

Next, we know that the gradient with respect to θ of integral of the probability is 0. We once again apply Leibniz rule and simplify

$$0 = \nabla_\theta \int_{z \in D_\theta} p(z|\theta) dz \tag{16}$$

$$= \nabla_\theta \int_{Q_{1-\alpha}(Z;\theta)}^{b} p(z|\theta) dz \tag{17}$$

$$= \int_{Q_{1-\alpha}(Z;\theta)}^{b} \nabla_\theta p(z|\theta) dz - p(Q_{1-\alpha}(Z;\theta)|\theta) \nabla_\theta Q_{1-\alpha}(Z;\theta) \tag{18}$$

Finally, combining equation 15 and 18 we cab simplify to the final equation.

$$\nabla_\theta J_{risk}(\theta;\alpha) = \frac{1}{\alpha} \int_{Q_{1-\alpha}(Z;\theta)}^{b} \nabla_\theta p(z|\theta)(z - Q_{1-\alpha}(Z;\theta)) dz \tag{19}$$

$$= E_{Z \sim p(Z|\theta)}[(Z - Q_{1-\alpha}(Z;\theta)) \nabla_\theta \log p(Z|\theta) | Z \geq Q_{1-\alpha}(Z;\theta)] \tag{20}$$

$$\nabla_\theta J_{risk}(\theta;\alpha) = E_{\tau \sim p(\tau|\theta)}[\nabla_\theta \log p(\tau|\theta) | R(\tau) > R_\alpha(\theta)] \tag{21}$$

This gives us a new risk seeking gradient for an unbiased risk seeking equation. The primary disadvantage with this policy is that is is no guarantee of convergence. However, since our method doesn't need to converge to the ground truth solution, this policy will be beneficial.

Current Results

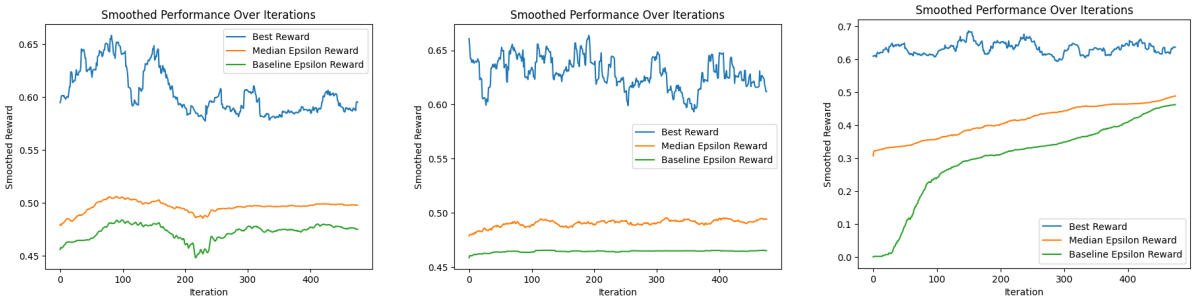
Table 1: Symbolic Regression Test Functions [4]

Name	Function	Variables	Complexity/Difficulty
Feynman 3_12_43	$(nh)/(2\pi)$	n, h	5
Feynman 3_17_37	$\beta(1 + \alpha\cos(\theta))$	α, β, θ	8
Nguyen_12	$x^4 - x^3 + \frac{1}{2}y^2 - y$	x, y	~ 12

In testing these three policies I selected three relatively simple SR problems (list in table 1) to be more easily able to understand the effects of the polices. The first problem is a solution that the actor will always discover the equation used to build the dataset. The second problem is slightly more difficult due to the dimensionality and the inclusion of the cosine function. Due to the simplicity of the problem the actor should be able to discover the function if it is able to explore and then exploit the space well. For both of these problems a single dataset was collected from 100 randomly sampled points and was tested only 1 time. Lastly, nguyen is a interesting problem that the original DSR paper failed to solve in all cases. While I have made other changes to the underlying model, I hope that this difficult problem will show discernible difference between the three policies. This problem we gave the methods 400 points from a grid. For all of the graphs below a smooth effect has been apply to make graphs interoperable.

For problem 1, I want to examine how these policies influence the equation's ability to exploit the space. Figure 1 shows how the actor learns to improve over 500 iterations. The primary difference between the DSR policy and the two new policies is that the two new policies do not drop in performance as iterations go on. The DSR policy seems to hit a wall around iteration 200 and shows a slight decrease in best reward performance. Overall, all three policies could find the ground truth equation and performed similarly. Lastly, note that the linear policy starts at a lower

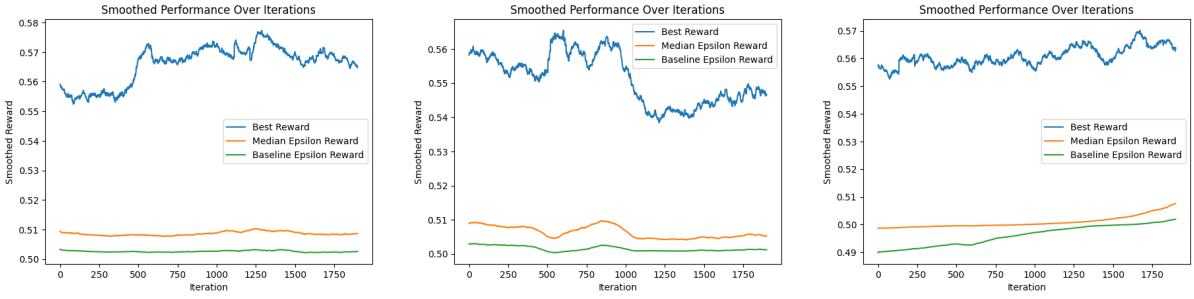
baseline and median due to having a large initial α value; however, iteration 500 seems to have a similar performance to that of the other policies.



(a) Standard Policy (b) Unbiased Policy (c) Linear Policy

Figure 1: Feynman 3_12_43 with Different Policies ($\epsilon = \alpha$)

Problem 2 was a lot more interesting as the unbiased policy could not find the ground truth equation, while the DSR policy and linear policy could. Furthermore, we see an inverse to problem 1 in the performance of the DSR and unbiased policies. In this problem, the DSR policy showed continual improvement until around iteration 1250, while the simplified policy had some improvement until 800, at which point its performance significantly decreased. Lastly, the linear policy seemed to have minimal change in performance other than maintaining a slow, continuous growth throughout all 2000 iterations. This continuous growth could signify that a tail barrier has yet to occur with the linear policy. A tail barrier could have occurred at iteration 1250 for the DSR policy. The unbiased policy seemed to have diverged away from the solution during cycles 900 to 1100. While it showed signs of recovering in the remaining iterations, this was not enough for it to discover the ground truth equation.

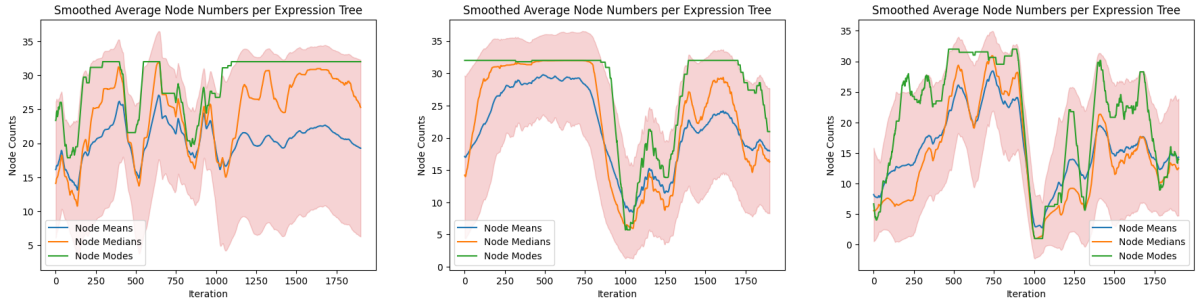


(a) Standard Policy (b) Unbiased Policy (c) Linear Policy

Figure 2: Feynman 3_17_37 with Different Policies ($\epsilon = \alpha$)

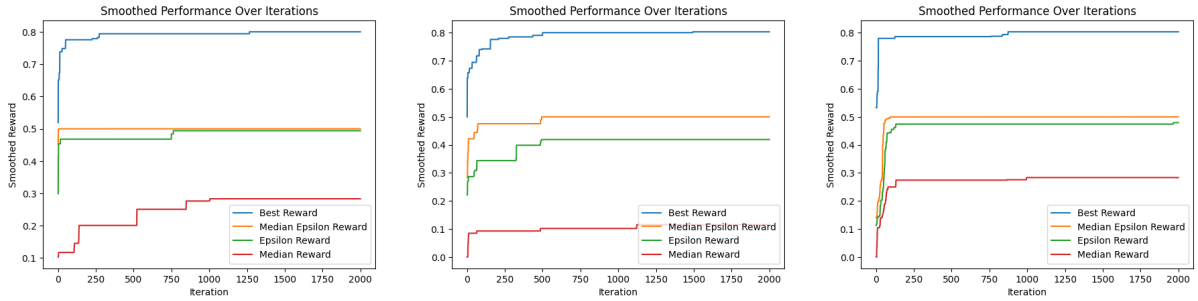
Problem 3 is by far the most challenging problem. None of the three policies managed to find the actual equation and all found an equation with an NMSE of 0.80. The standard performance graphs were not very insightful for these runs. Instead, we have two graphs for each policy, one showing the average size of the equation in each epoch and one showing the cumulative performance of the models. First, looking at the standard policy, we can see that the node counts are all over the place, and after around 1000 iterations, the variance, red region, of the predictions encompassed the entire space. Furthermore, if we look at the cumulative performance, we can see a tail barrier occurred near 1000 iterations because the median epsilon reward becomes equivalent to the epsilon reward. The standard policy could only learn slightly after this point, leading to significant variance.

The unbiased policy maintained a tighter variance of equations predicted throughout the run. Furthermore, we know that a tail barrier cannot form, which is further shown by the median epsilon reward, never touching the epsilon reward. Lastly, the linear policy performs very similarly to the standard policy, with the primary exception of not finding a tail barrier. The lack of tail barriers in the linear policy aligns with the mathematical backing in the previous section, showing the decrease in the probability of encountering one. However, the final performance is not statically different from any of the other policies. Overall, the linear policy is performing the best by avoiding tail barriers while maintaining the standard policy's performance.



(a) Standard Policy (b) Unbiased Policy (c) Linear Policy

Figure 3: Nguyen 12 Expression Tree Predictions with Different Policies ($\epsilon = \alpha$)



(a) Standard Policy (b) Unbiased Policy (c) Linear Policy

Figure 4: Nguyen 12 Cumulative Performance with Different Policies ($\epsilon = \alpha$)

Future Work

Future work on this problem should be spent using an environment that is more understandable to humans and has a simpler actor. For example, a Markov decision process can be used with a path-finding environment. However, the goal of only discovering one instance of the best solution would still need to be enforced to allow the unbiased method to be valid. By implementing this in a more understandable environment with a similar actor, the differences in the policy will become more understandable.

References

[1] B. K. Petersen, M. Landajuela, T. N. Mundhenk, C. P. Santiago, S. K. Kim, and J. T. Kim. "Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients," arXiv.org. (Dec. 10, 2019), [Online]. Available: https://arxiv.org/abs/1912.04871v4 (visited on 10/10/2023).

[2] P.-A. Kamienny, S. d'Ascoli, G. Lample, and F. Charton, "End-to-end symbolic regression with transformers," presented at the Advances in Neural Information Processing Systems, May 16, 2022. [Online]. Available: https://openreview.net/forum?id=GoOuIrDHG_Y (visited on 10/10/2023).

[3] I. Greenberg, Y. Chow, M. Ghavamzadeh, and S. Mannor, *Efficient risk-averse reinforcement learning*, Oct. 12, 2022. DOI: 10.48550/arXiv.2205.05138. arXiv: 2205.05138[cs]. [Online]. Available: http://arxiv.org/abs/2205.05138 (visited on 03/15/2024).

[4] G. Bomarito, P. Leser, N. Strauss, K. Garbrecht, and J. Hochhalter, "Automated learning of interpretable models with quantified uncertainty," *Computer Methods in Applied Mechanics and Engineering*, vol. 403, p. 115 732, Jan. 1, 2023. DOI: 10.1016/j.cma.2022.115732.