Zipf's Law and CCDF Analysis on Deep Reinforcement Learning

Mioto Takahashi[1, *]

[1] *UVM CSDS MS*

(Dated: December 16, 2023)

With the recent surge in research regarding deep reinforcement learning, which utilizes black-box statistical models, it is important to have methods of intuitive analysis and evaluation. The project focuses on analyzing the state space of deep reinforcement learning controllers during training using Zipf and comparative cumulative distribution functions. Our results indicate that there is no change in the behavior exhibited by snapshots of a deep neural network in the learning stage when it comes to the exponents in the scaled linear regression for both distributions.

I. Introduction

Control systems that utilize deep neural networks or other black-box statistical models, mainly deep reinforcement learning (DRL), has been studied extensively in recent years. While a multitude of methods of DRL have been proposed, the lack of interpretability that comes from the usage of black-box statistical models remains to be an open problem.

Reinforcement learning (RL) is a control system which learns to complete a task, such as moving a cart as far as possible on the x-axis. An RL system is composed of a policy, which indicates the controller of the system such as those that dictate the velocity of the cart, and an environment, which indicates the description of the cart, such as its position and velocity. In discrete-time systems, such as physics engines, both the policy and the environment output values per timestep. The environment outputs values referred to as the state, such as the position and velocity of the cart in the current timestep. The state is then fed to the policy, which outputs values called action, such as the change in the velocity of the cart. The action is then fed to the environment, which causes it to alter the value of the state in the succeeding timestep. Additionally, the environment outputs a value referred to as the reward, which indicates the task the RL policy is attempting to complete. In the context of moving a cart on the x-axis, this would be the current position within the x-axis or the change in the position compared to the previous timestep. In DRL, a deep neural network that inputs a state and outputs an action, is used for the policy. The policy is updated to maximize the cumulative reward, or the return, using metaheuristic algorithms, such as evolutionary algorithms.

Research in DRL have revealed that the maximization of the mutual information between the actions and the resulting changes in the state can aid in having the policies learn to execute manually defined tasks with higher performance [1]. Subfields of DRL, such as meta RL, have managed to have the policy learn a multitude of similar tasks, such as solving various mazes [2]. Another subfield, hierarchical RL, multiple tasks, such as jumping and running, were able to be encoded in a single deep neural network without the usage of any manual definition of the type of task to perform [3].

However, with the low interpretability of mutual information, analysis of DRL policies become difficult. Directly investigating data points in trajectories of the policies using more interpretable statistical models, such as linear regression, has not been effective so far to our knowledge.

This project aims to utilize Zipf distribution, complementary cumulative distribution function (CCDF), and exponentially scaled linear regression to gain more intuitive understanding of the trajectories of DRLs in the process of learning tasks. More specifically the question regarding, given a predefined deep neural network architecture, how would the exponentials and r-squared values of linear regression models over plots of the state space alter over the learning process, or epochs.

II. Experiment

A. Data Generation

The OpenAI Gym environment [4] "CartPole-v1", a 2-dimensional physics simulation of the inverted pole-balancing problem with a controllable cart, is used for the environment of the DRL (Figure 1). The cart control is binary, where only left and right movements per frame are allowed. The state space is composed of the cart's position, velocity, the pole's angle with

* mtakaha1@uvm.edu

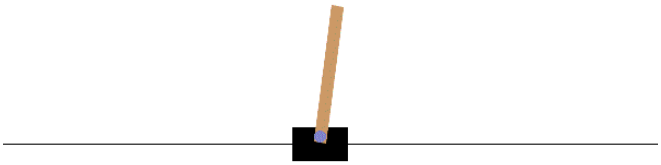


FIG. 1. OpenAI Gym "Cartpole-v1" Environment [4]

respect to the vertical axis, and angular velocity, all of which are floating point values. The reward function, generated per timestep, is a binary value that indicates whether the pole is staying upright based on whether the absolute value of the pole's angle is above or below the threshold of 0.475 radians.

For the DRL policy, a deep neural network with 4-dimensional input neurons, 16-dimensional hidden neurons, and 2 output neurons is defined, where the activation function for the 2 output neurons are softmax functions and the rest are hyperbolic tangent functions.

An evolutionary algorithm [5] with mutation and tournament selection, and without crossover and novelty search, is used for the training process for 1000 epochs. All of the values in each genome, which is merely the weight values in the deep neural network stretched to a single array, are binary. The best performing deep neural network policy in the gene pool generates the trajectory of the state space for 1000 timesteps.

For the fitness function of the evolutionary algorithm, the sum of the reward for the "CartPole-v1" problem generated each timestep without any reward decay proportional to the time horizon is used. A return value calculated with reward decay taken into consideration is as, $R = \sum_{t=0}^{T-1} \delta^t r_t$, where r_t is the reward at timestep t, R is the return, T is the total timesteps in the simulation, and δ is the reward decay rate. While it is commonplace to use such in DRL, this is not taken into consideration in this work for simplicity and the lack of necessity due to the difficulty of the problem.

In terms of the number of parent genomes μ and child genomes λ in the $\mu + \lambda$ evolutionary strategy, $\lambda = 10, \mu = 10$.

B. Evaluation

Since the chances of floating point values appearing 2 or more times in the trajectories would be extremely low, every value in it is rounded to the hundredths. The frequency f of each visited state r is plotted as a Zipf distribution, in which exponentially scaled linear regression is used to fit to. Additionally, the frequency k and the number of visited state spaces with the same frequency $N_{\geq k}$ is plotted as a CCDF, in which, similar to the Zipf distribution, is fit to exponentially scaled linear regression models.

The Zipf distribution and CCDF's scaling coefficients γ and α are

$$f = \alpha_{Zipf} r^{-\alpha} + \beta_{Zipf} \tag{1}$$

$$N_{\geq k} = \alpha_{CCDF} k^{1-\gamma} + \beta_{CCDF} \tag{2}$$

where α and β are the slope and the intercept.

For both the epoch-number of linear regression models in Zipf distribution and CCDF, the change in the best-fitting exponential values $-\alpha$ and $1-\gamma$ over epochs is quantitatively examined, on whether it would produce monotonically increasing, monotonically decreasing, concave, convex, or other geometry.

Additionally, how well γ and α satisfies

$$\alpha = \frac{1}{\gamma - 1} \tag{3}$$

$$\alpha(\gamma - 1) = 1 \tag{4}$$

is evaluated.

III. Results

The fitness curve for the deep neural network is as figure 2. A solution that is able to balance the pole for all 1000 timesteps is found at the 21st epoch. Occasional dips in the curve after the 21st epoch can be observed.

The diversity, or the standard deviation of the genomes, over the epochs is as figure 3. While the expected diversity curve in learning algorithms is a monotonical decrease, due to the notion that similar solutions would be left in the gene pool as the epoch progresses, the results show otherwise.

The Zipf distribution and CCDF for the visited state or observation space of the agent, whose counting scheme

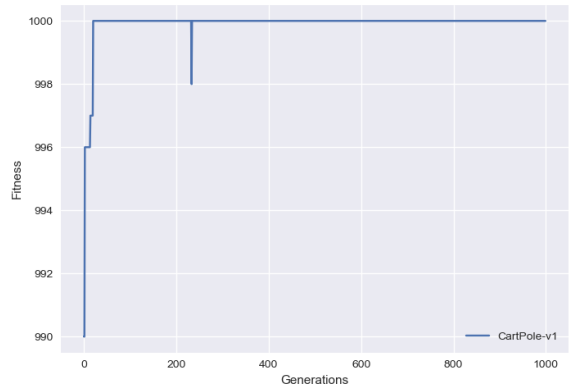


FIG. 2. Fitness Curve

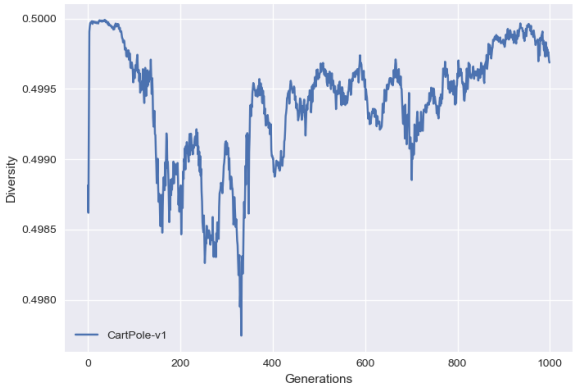


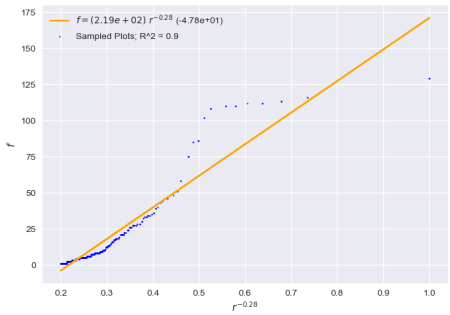
FIG. 3. Diversity Curve

is specified in section II B is scaled and linearly regressed in figure 4. Both plots for the intermediate epochs are in the provided code.

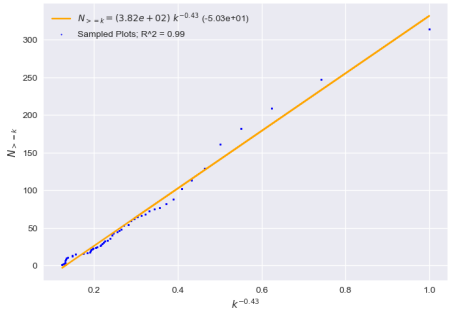
The scaling coefficients γ and α and the left-hand side of equation 4 over the 1000 epochs is plotted in figure 5. Equation 4 is only satisfied as high as 0.3871. The variances of γ and α are high, and it is therefore trivial that there is no correlation between such values and the training epochs.

IV. Discussion

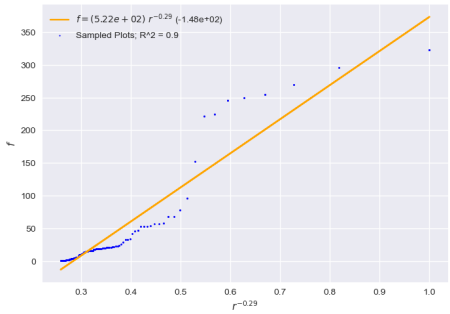
DRL's training using evolutionary algorithm on the "CartPole-v1" is a success. The occasional dips in the fitness curve can be a result of rare initial conditions of the cart and the pole being more difficult than the others. Performance improvement can be achieved if floating point values are used for the weight values in



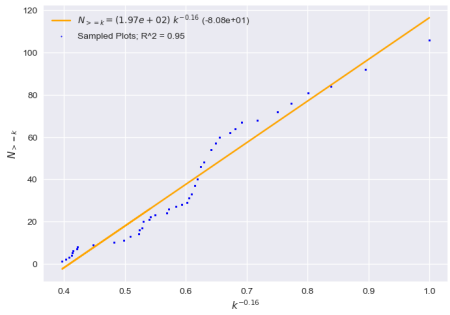
(a) Zipf Distribution at Epoch 0



(b) CCDF at Epoch 0



(c) Zipf Distribution at Epoch 1000



(d) CCDF at Epoch 1000

FIG. 4. Power-Scaled Linear Regression. The scaling coefficient and the r-squared values are indicated in the legend in each plot.

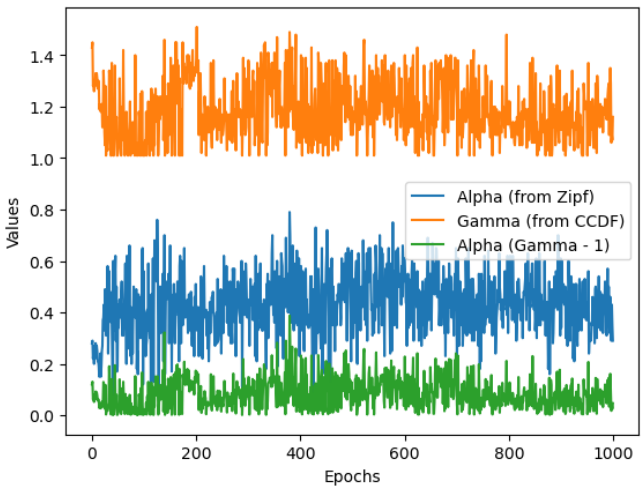


FIG. 5. Zipf distribution's scaling coefficient γ, CCDF's scaling coefficient α, and $\alpha(1-\gamma)$ over 1000 epochs

the deep neural network governing the agent instead of bits. The rise in the diversity may be due to the evolutionary algorithm finding more solutions that can successfully balance the pole on the cart for 1000 timesteps through tournament selection.

For each epoch, the r-squared value of the CCDF distribution is higher than those of Zipf distribution, indicating that the former resembles more like a power law distribution than the latter (figure 4).

The shape of the curves in figure 4 composed of the points in both the Zipf distribution and CCDF seem to have a concave component and a convex component. Combined with the result which indicate that the data do not fit equation 4 with much accuracy, it may be viable to conduct regression with functions that are of higher order than linear.

The changes in the scaling coefficients γ and α do not have correlation between the number of generations or epochs. While integration of mutual information in the reward function could be used to aid DRL in preliminary research [2, 3], our results show that the same cannot be said for power laws.

Our work conducted analysis using scaled linear regression for Zipf distribution and CCDF composed of state or observation spaces visited by DRL agents throughout their training process. We found that the power law coefficients γ and α for the resulting distribution do not satisfy properties found in such that can be regressed to linear functions. Additionally, there were no correlation in the changes of both coefficients and the number of generations, indicating the lack of usefulness in incorporating power law to reward functions for possible boost in learning speed. Further research connecting power law and DRL could have the Zipf distribution and CCDF fit to high-order functions.

V. Code

The supplementary code for the project is on

https://github.com/johnlime/pocs_paper

[1] K. Frans, J. Ho, X. Chen, P. Abbeel, and J. Schulman, Meta Learning Shared Hierarchies, (2017), arXiv:1710.09767.
[2] S. Mohamed and D. J. Rezende, Variational information maximisation for intrinsically motivated reinforcement learning, Advances in Neural Information Processing Systems **2015-January**, 2125 (2015), arXiv:1509.08731.
[3] B. Eysenbach, A. Gupta, J. Ibarz, S. Levine, A. Gupta, S. Levine, J. Ibarz, and S. Levine, Diversity is All You Need: Learning Skills without a Reward Function, 7th International Conference on Learning Representations, ICLR 2019 , 1 (2018), arXiv:1802.06070.
[4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, OpenAI Gym, , 1 (2016), arXiv:1606.01540.
[5] L. G. M. U. Sean, *Essentials of Metaheuristics: A Set of Undergraduate Lecture Notes* (Optimization, 2010) pp. 1–220.