Evolutionary Strategies: A Simple and Often-Viable Alternative to Reinforcement Learning

Ashwin Rao

ICME, Stanford University

November 9, 2019

Introduction to Evolutionary Strategies

- Evolutionary Strategies (ES) are a type of Black-Box Optimization
- Popularized in the 1970s as *Heuristic Search Methods*
- Loosely inspired by natural evolution of living beings
- We focus on a subclass called Natural Evolution Strategies (NES)
- The original setting was generic and nothing to do with MDPs or RL
- Given an objective function $F(\psi)$, where ψ refers to parameters
- We consider a probability distribution $p_\theta(\psi)$ over ψ
- Where θ refers to the parameters of the probability distribution
- We want to maximize the average objective $\mathbb{E}_{\psi \sim p_\theta}[F(\psi)]$
- We search for optimal θ with stochastic gradient ascent as follows:

$$\nabla_\theta(\mathbb{E}_{\psi \sim p_\theta}[F(\psi)]) = \nabla_\theta(\int_\psi p_\theta(\psi) \cdot F(\psi) \cdot d\psi)$$

$$= \int_\psi \nabla_\theta(p_\theta(\psi)) \cdot F(\psi) \cdot d\psi = \int_\psi p_\theta(\psi) \cdot \nabla_\theta(\log p_\theta(\psi)) \cdot F(\psi) \cdot d\psi$$

$$= \mathbb{E}_{\psi \sim p_\theta}[\nabla_\theta(\log p_\theta(\psi)) \cdot F(\psi)]$$

NES applied to solving Markov Decision Processes (MDPs)

- We set $F(\cdot)$ to be the (stochastic) *Return* of a MDP
- ψ refers to the parameters of a policy $\pi_\psi : \mathcal{S} \rightarrow \mathcal{A}$
- ψ will be drawn from an isotropic multivariate Gaussian distribution
- Gaussian with mean vector θ and fixed diagonal covariance matrix $\sigma^2 I$
- The average objective (*Expected Return*) can then be written as:

$$\mathbb{E}_{\psi \sim p_\theta}[F(\psi)] = \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}[F(\theta + \sigma \cdot \epsilon)]$$

- The gradient (∇_θ) of *Expected Return* can be written as:

$$\mathbb{E}_{\psi \sim p_\theta}[\nabla_\theta (\log p_\theta(\psi)) \cdot F(\psi)]$$

$$= \mathbb{E}_{\psi \sim \mathcal{N}(\theta, \sigma^2 I)}[\nabla_\theta (\frac{-(\psi - \theta)^T \cdot (\psi - \theta)}{2\sigma^2}) \cdot F(\psi)]$$

$$= \frac{1}{\sigma} \cdot \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}[\epsilon \cdot F(\theta + \sigma \cdot \epsilon)]$$

A sampling-based algorithm to solve the MDP

- The above formula helps estimate gradient of *Expected Return*
- By sampling several ϵ (each ϵ represents a *Policy* $\pi_{\theta+\sigma\cdot\epsilon}$)
- And averaging $\epsilon \cdot F(\theta + \sigma \cdot \epsilon)$ across a large set (n) of ϵ samples
- Note $F(\theta + \sigma \cdot \epsilon)$ involves playing an episode for a given sampled ϵ, and obtaining that episode's *Return* $F(\theta + \sigma \cdot \epsilon)$
- Hence, n values of ϵ, n *Policies* $\pi_{\theta+\sigma\cdot\epsilon}$, and n *Returns* $F(\theta + \sigma \cdot \epsilon)$
- Given gradient estimate, we update θ in this gradient direction
- Which in turn leads to new samples of ϵ (new set of *Policies* $\pi_{\theta+\sigma\cdot\epsilon}$)
- And the process repeats until $\mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}[F(\theta + \sigma \cdot \epsilon)]$ is maximized
- The key inputs to the algorithm will be:
  - Learning rate (SGD Step Size) α
  - Standard Deviation σ
  - Initial value of parameter vector θ_0

The Algorithm

Algorithm 0.1: NATURAL EVOLUTION STRATEGIES$(\alpha, \sigma, \theta_0)$

for $t \leftarrow 0, 1, 2, \ldots$

do $\begin{cases} \text{Sample } \epsilon_1, \epsilon_2, \ldots \epsilon_n \sim \mathcal{N}(0, I) \\ \text{Compute Returns } F_i \leftarrow F(\theta_t + \sigma \cdot \epsilon_i) \text{ for } i = 1, 2, \ldots, n \\ \theta_{t+1} \leftarrow \theta_t + \frac{\alpha}{n\sigma} \sum_{i=1}^{n} \epsilon_i \cdot F_i \end{cases}$

Resemblance to Policy Gradient?

- On the surface, this NES algorithm looks like Policy Gradient (PG)
- Because it's not Value Function-based (it's Policy-based, like PG)
- Also, similar to PG, it uses a gradient to move towards optimality
- But, ES does not interact with the environment (like PG/RL does)
- ES operates at a high-level, ignoring (state,action,reward) interplay
- Specifically, does not aim to assign credit to actions in specific states
- Hence, ES doesn't have the core essence of RL: *Estimating the Q-Value Function of a Policy and using it to Improve the Policy*
- Therefore, we don't classify ES as Reinforcement Learning
- We consider ES to be an alternative approach to RL Algorithms

ES versus RL

- Traditional view has been that ES won't work on high-dim problems
- Specifically, ES has been shown to be data-inefficient relative to RL
- Because ES resembles simple hill-climbing based only on finite differences along a few random directions at each step
- However, ES is very simple to implement (no Value Function approx. or back-propagation needed), and is highly parallelizable
- ES has the benefits of being indifferent to distribution of rewards and to action frequency, and is tolerant of long horizons
- This paper from OpenAI Researchers shows techniques to make NES more robust and more data-efficient, and they demonstrate that NES has more exploratory behavior than advanced PG algorithms
- I'd always recommend trying NES before attempting to solve with RL