MATH 305

Hyperparameter Optimization Using Reinforcement Learning

Alican Akca -Yusuf Erol

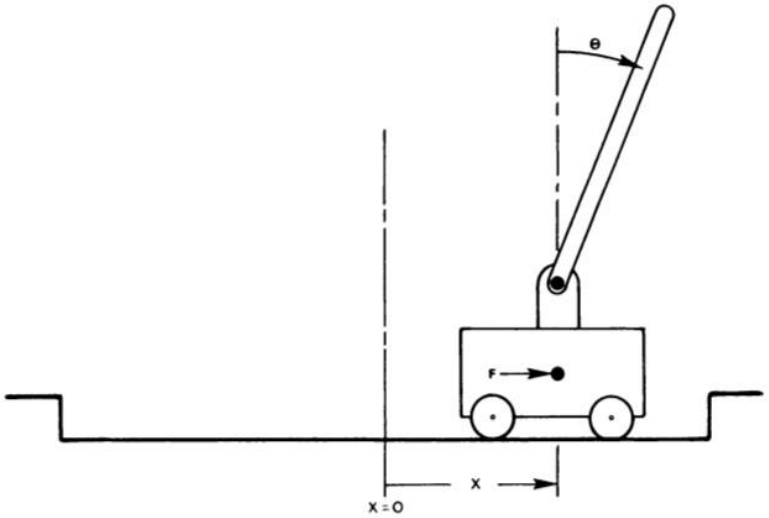
Content

METHODOLOGY

Methodology:

Environment

The goal is to train an agent to balance a pole on a cart by moving the cart left or right.



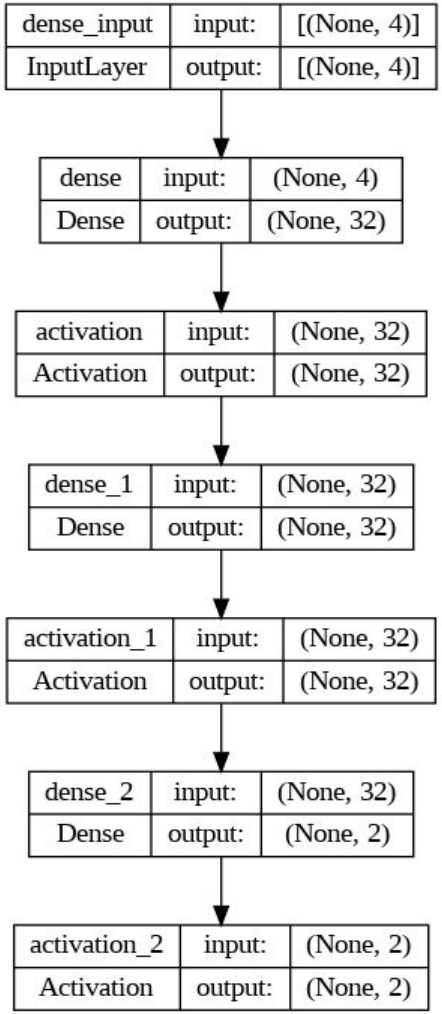
Action Space	Discrete(2)
Observation Shape	(4,)
Observation High	[4.8 inf 0.42 inf]
Observation Low	[-4.8 -inf -0.42 -inf]
Import	`gym.make("CartPole-v1")`

Methodology:

Agent

The agent makes decisions by choosing actions that are expected to maximize the cumulative reward over time.

The agent is a neural network model that takes the state of the environment as input and outputs the action to be taken.



Methodology:

Optimization Problem

The objective function is to minimize the validation loss (val_loss) which is the Mean Squared Error (MSE).

*Given a set of **n** samples, where for each sample **i**, the predicted value is **y^i** and the actual value is **yi**, the MSE is calculated as:*

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

Methodology:

Rewards and Penalties

In the context of reinforcement learning, Q-values that must be maximized represent the expected future reward for taking a certain action in a certain state.

$$Q(s,a) = r + \gamma \max_{a'} Q(s', a')$$

where:

- s is the current state,
- a is the action taken,
- r is the immediate reward received after taking action a in state s,
- s′ is the new state after taking action a,
- a′ is the action taken in state s′,
- γ is the discount factor which determines the present value of future rewards.

Algorithm

Algorithm

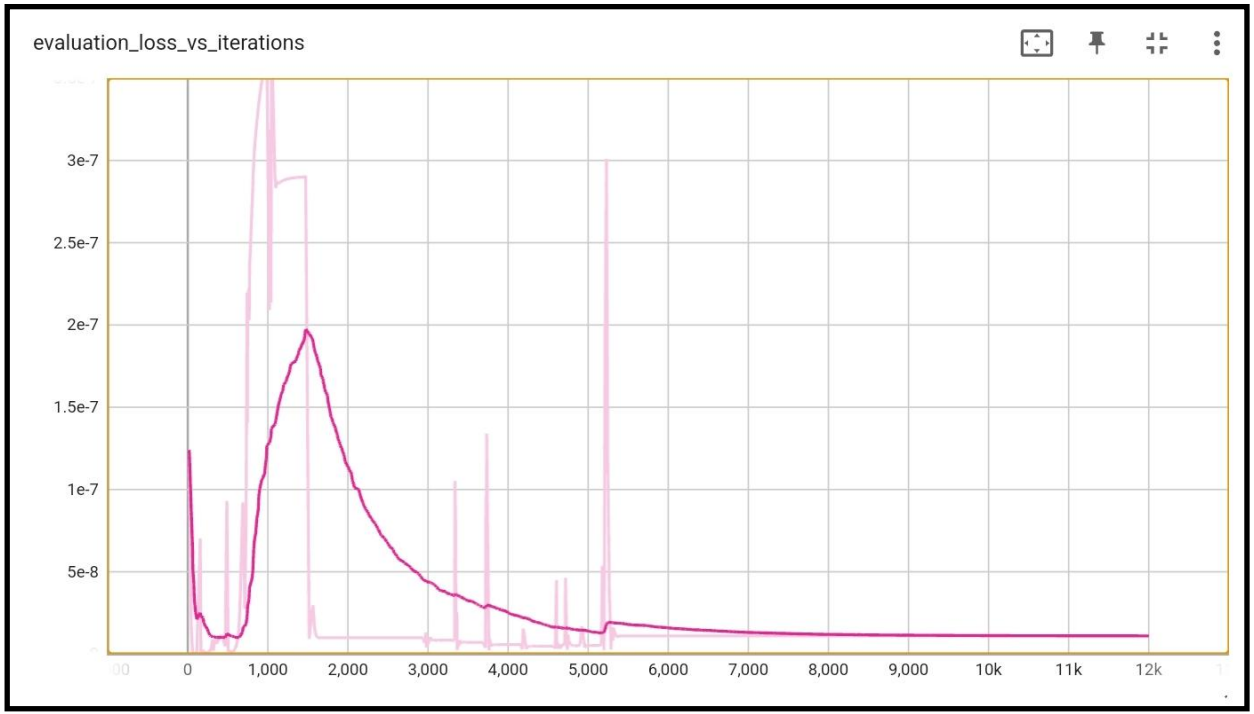
Let f: $\mathbb{R}^n \rightarrow \mathbb{R}$ be the fitness or cost function which must be minimized. Let $\mathbf{x} \in \mathbb{R}^n$ designate a position or candidate solution in the search-space. The basic RandomSearch algorithm can then be described as:

1. Initialize **x** with a random position in the search-space.
2. Until a termination criterion is met (e.g. number of iterations performed, or adequate fitness reached), repeat the following:
    1. Sample a new position **y** from the hypersphere of a given radius surrounding the current position **x** (see e.g. Marsaglia's technique for sampling a hypersphere.)
    2. If $f(\mathbf{y}) < f(\mathbf{x})$ then move to the new position by setting $\mathbf{x} = \mathbf{y}$

Evaluation of the Inferences

Evaluation of the Inferences

- Stability
- Efficiency
- Robustness
- Generalization



References

[1] Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems ANDREW G. BARTO, Member, IEEE, Richard S. Sutton, and Charles w. Anderson (0018-9472/83/0900-083401.00 01983 IEEE)

Thanks for listening!