AWS DeepRacer

Reinforcement Learning - CSE 4/510 Checkpoint

5/5/2020

Team Members:
Raunaq Jain (50320026)
Bipul Kumar (50320998)
Joseph Naro (50286981)

University at Buffalo The State University of New York

Introduction

AWS DeepRacer is a platform to teach reinforcement learning. To start with, models are trained on simulated tracks. The user can create their own reward functions and choose the hyperparameters of the network. Once satisfactory results are obtained, the model is then transferred to the car.

The AWS DeepRacer car is an autonomous 1/18th scale race car designed to test reinforcement learning models by racing around on a physical track. It uses cameras and other sensors to view the track and objects and a reinforcement model to control the throttle and steering. The car shows how a model trained in a simulated environment can be transferred to the real world.

To play around, AWS DeepRacer provides various tracks and 3 different racing formats - Time Trial, Object Avoidance and Head-to-Head.

Description

- We wanted to know how DeepRacer works and how to create better DeepRacer models
- We also wanted to run the simulation locally
- Things we experimented with
  - Reward functions
  - Optimizations functions
  - Action space

Background 2

Community members worked to enable users to train DeepRacer locally. These allow people to train models without the costs of using AWS.

Running locally: (chosen approach)

aws-robomaker-sample-application-deepracer

- Installing ROS Kinetic: To run the simulation locally
- Gazebo: Will automatically get installed if installed ROS desktop full in prior steps
- AWS account setup

AWS Service Charges	$81.34
CloudWatch	$0.00
Data Transfer	$0.00
Key Management Service	$0.00
Kinesis Video Streams	$0.28
Lambda	$0.00
RoboMaker	$58.72
SageMaker	$16.05
Simple Storage Service	$0.31
Taxes	
US Sales Tax to be collected	$5.98

Implementation 1

- Over a dozen models tested
  - Only using PPO algorithm on AWS console
  - Most using 30 degrees as a maximum turning angle
- All models were trained on the re:Invent 2018 track
- Two different scenarios -
  - Time Trials
  - Head-to-Head Racing

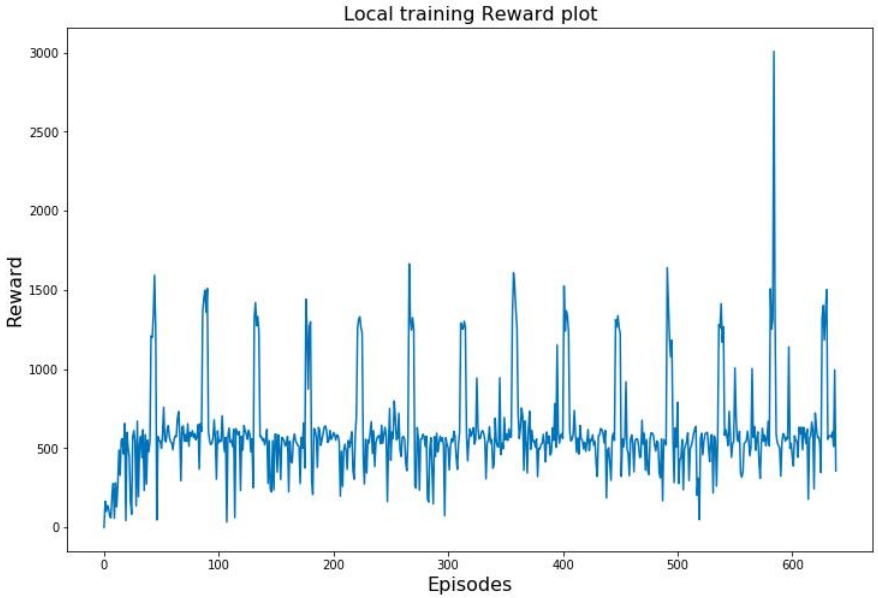
Training in Progress

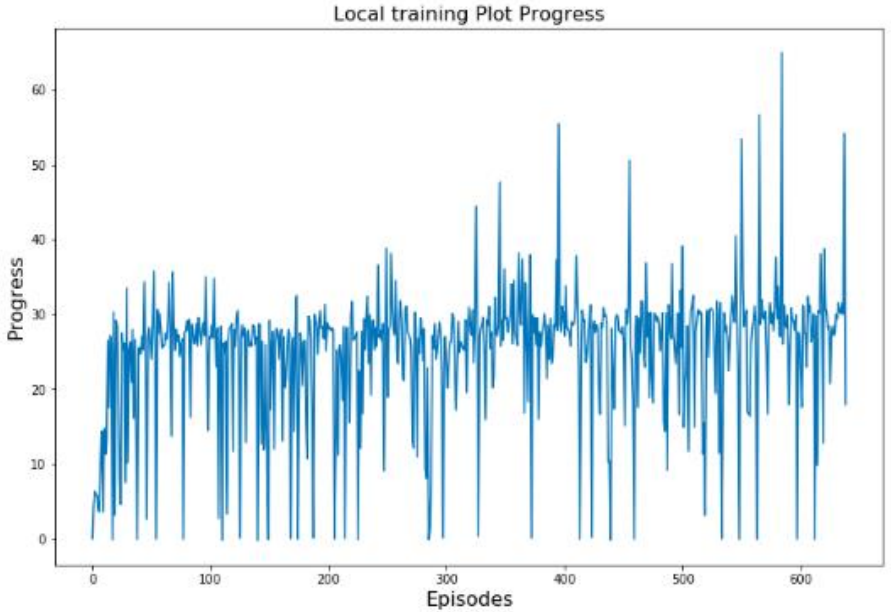
Visualize training of the blue car over time.

Implementation 2

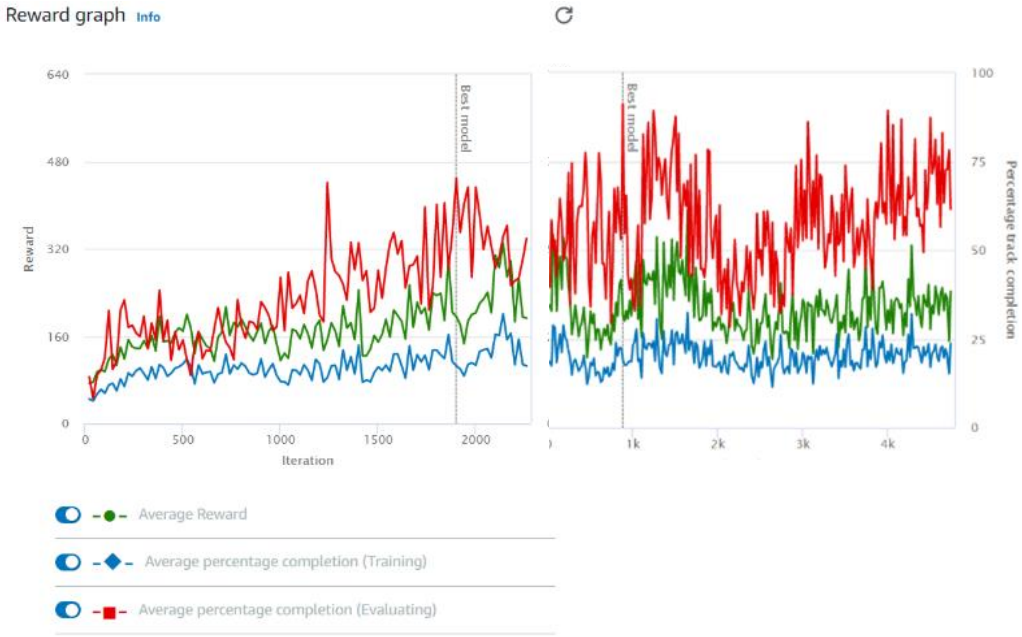
- AWS console only allows learning on PPO
- By setting system locally, cost can be minimized
- The environment and track are similar to the AWS console with more in our hands in terms of parameter flexibility.
- We documented implementing Rainbow DQN to compare it against PPO

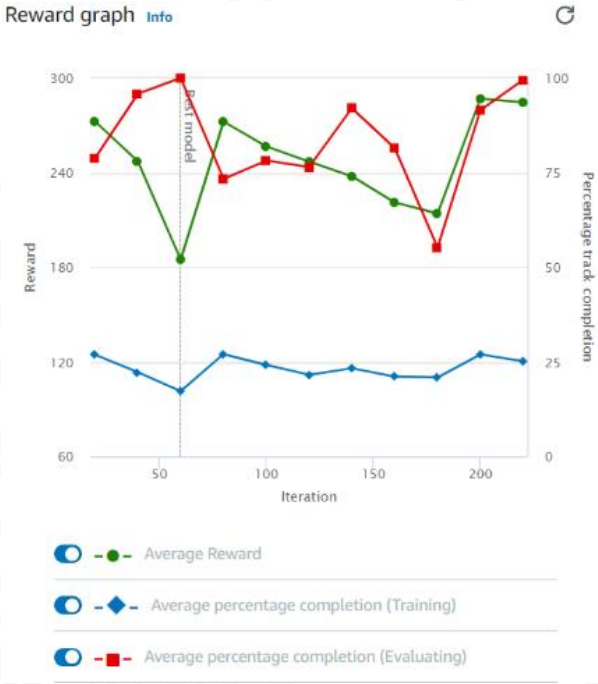
Results 1


Local training Reward plot
Reward
Episodes
0
500
1000
1500
2000
2500
3000
0
100
200
300
400
500
600


Local training Plot Progress
Progress
Episodes
0
10
20
30
40
50
60
0
100
200
300
400
500
600

Results 2


Reward graph
Info
Reward
Iteration
Best model
Percentage track completion
Average Reward
Average percentage completion (Training)
Average percentage completion (Evaluating)


Reward graph
Info
Reward
Iteration
Best model
Percentage track completion
Average Reward
Average percentage completion (Training)
Average percentage completion (Evaluating)

Results 3

Evaluation results

Trial	Time	Trial results (% track completed)	Status
1	00:00:01.496	5%	Off track
2	00:00:15.325	100%	Lap complete
3	00:00:15.219	100%	Lap complete

Key observations / summary

- Reducing the range of possible actions can improve performance
- Alternative optimization algorithms show promise
- DeepRacer available on AWS is limited and costly
- Running on local is still not properly documented (which we hope to change_

Thank you

Picture: https://medium.com/proud2becloud/deepracer-our-journey-to-the-top-ten-257ff69922e