Lab 1: Reward Function Constructors Championship!

Overview

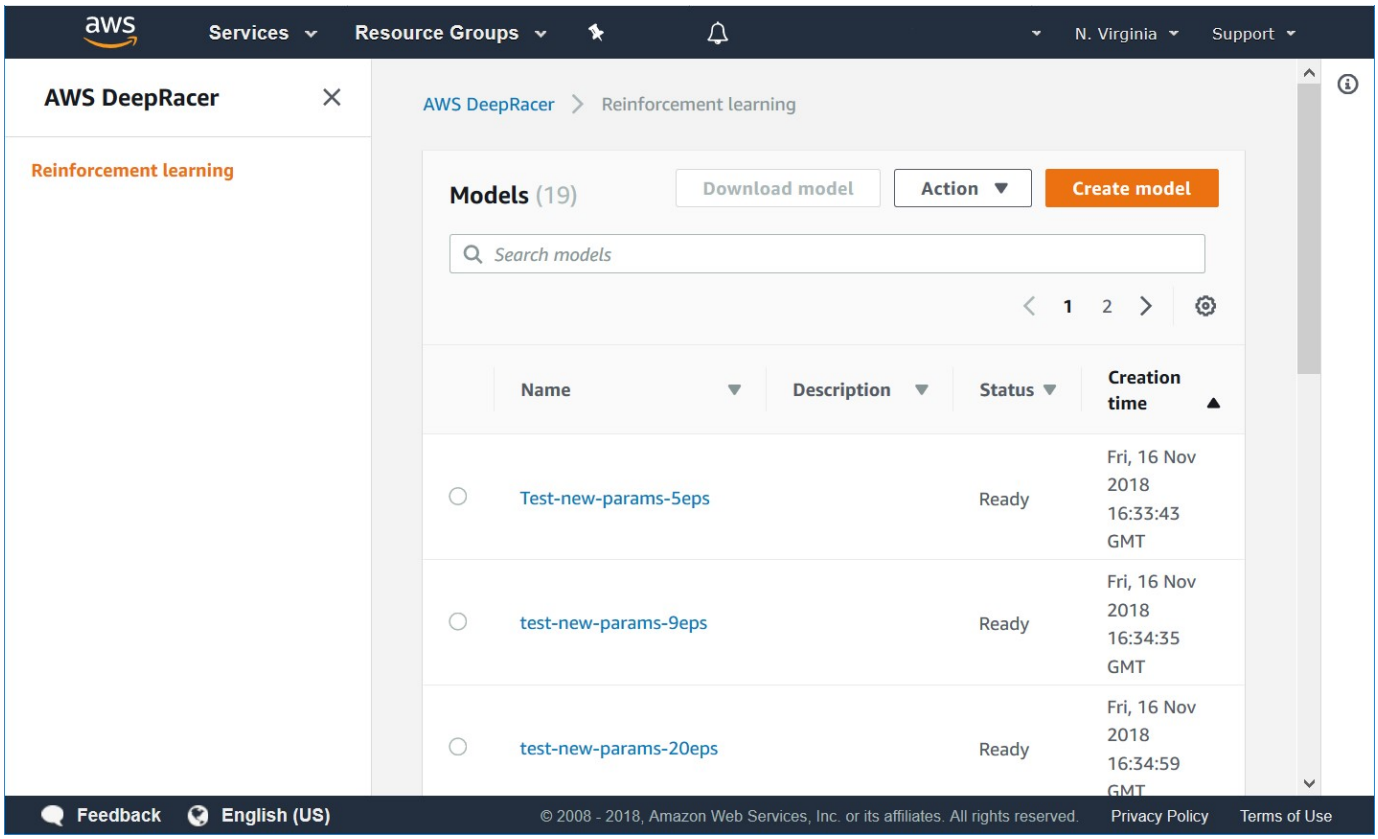
In this lab we will provide an overview of the AWS DeepRacer simulator found in the AWS Console. You will also get hands-on with the reward functions we use in AWS DeepRacer. You will form a team with fellow attendees at your table to take part in the **Reward Function Constructors Championship**.

Step 1: AWS Console

Logon to the AWS Console using the account details provided to you. Navigate to AWS DeepRacer, https://aws.amazon.com/deepracer (https://aws.amazon.com/deepracer). From the AWS DeepRacer landing page, expand the pane on the left and select **Reinforcement learning.**

Step 2: Model List Page

This page gives a list of all the models you have created and each model's respective status.



If you don't have any models this list will be empty, and you can create a model by choosing **Create model**. Once you have created models you can use this page to view the status of the model, for example is it training, or ready to download or evaluate. You can click on the model's name to proceed to the Model details page. Alternatively selecting the radio button next to a model name will allow you todelete the model, clone the model, or download the model, if training has completed.

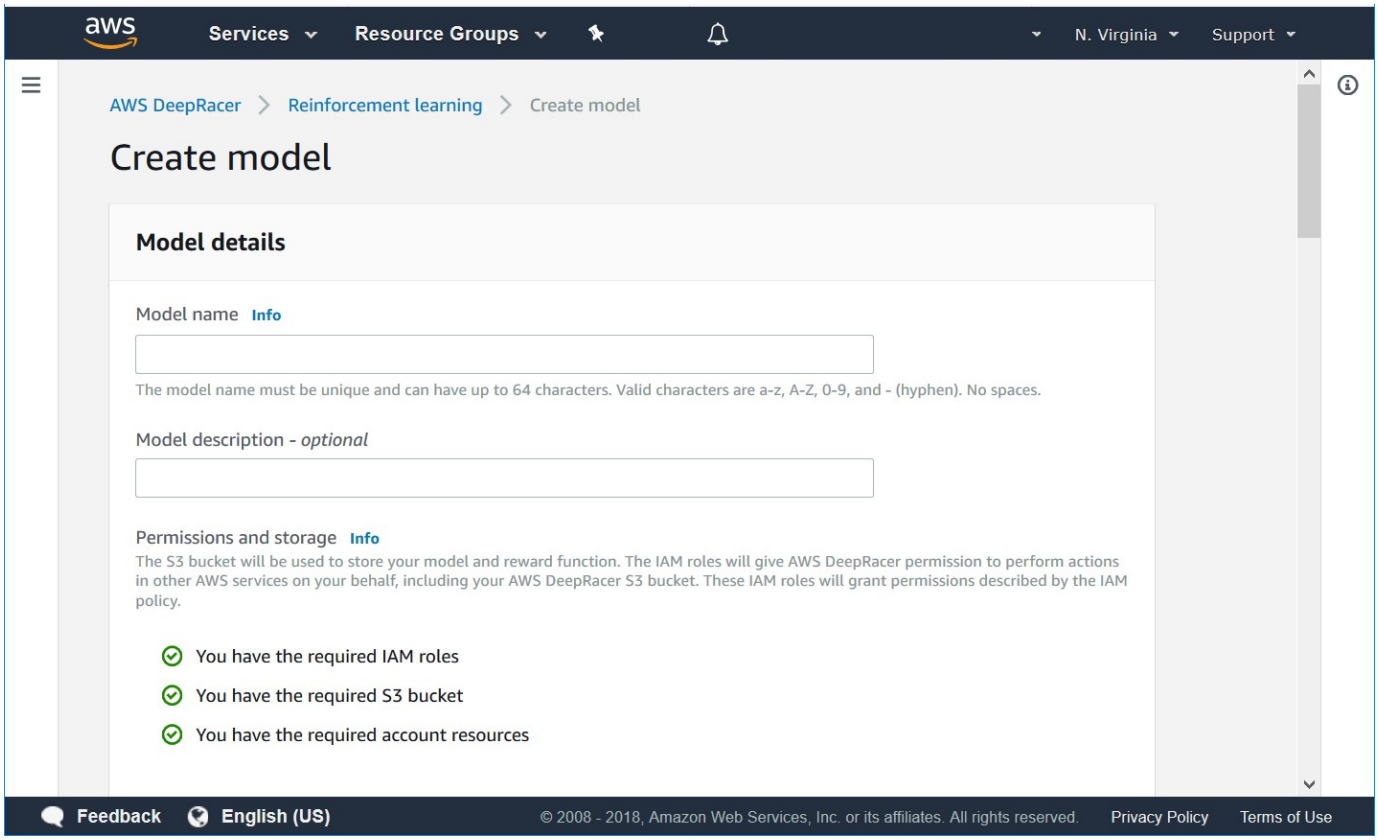
Choose **Create model**

Step 3: Create model

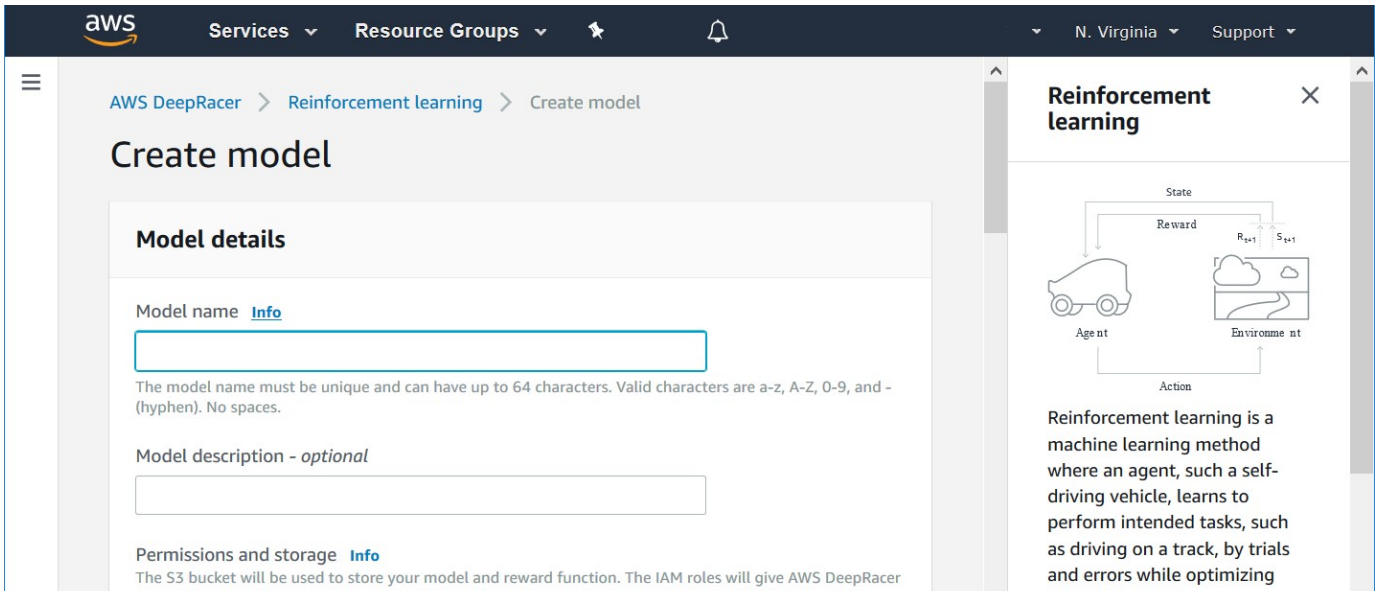
This page gives you the ability to create an RL model for AWS DeepRacer and start training. There are a few sections on the page, so let's look at each in detail.

Model details

Here you have the ability to name your model, provide a model description, and create the resources needed to start AWS DeepRacer training jobs. Note that the resources should be ready in the account given to you. If you do not see three green radio boxes beneath the Permission and storage sectino please let us know.



Info **Buttons** Throughout the console you will see **Info** buttons. When selected they will slide out an information pane from the right of the window, without navigating away from the page.



Step 4: Reward Function Constructors Championship

We are now going to run a short , hence the 10 minute max. time, excercise to help build your understanding of the reward function. We call this the Reward Function Constructors Championship. The goal of the championship is to see which team can in 15 minutes train a model that completes the largest percentage of the track in three evaluation runs.

Instructions

Form a team with the other attendees at your table.nFor the challenge you will have 10 minutes to come up with a reward function and 15 minutes to train the model.

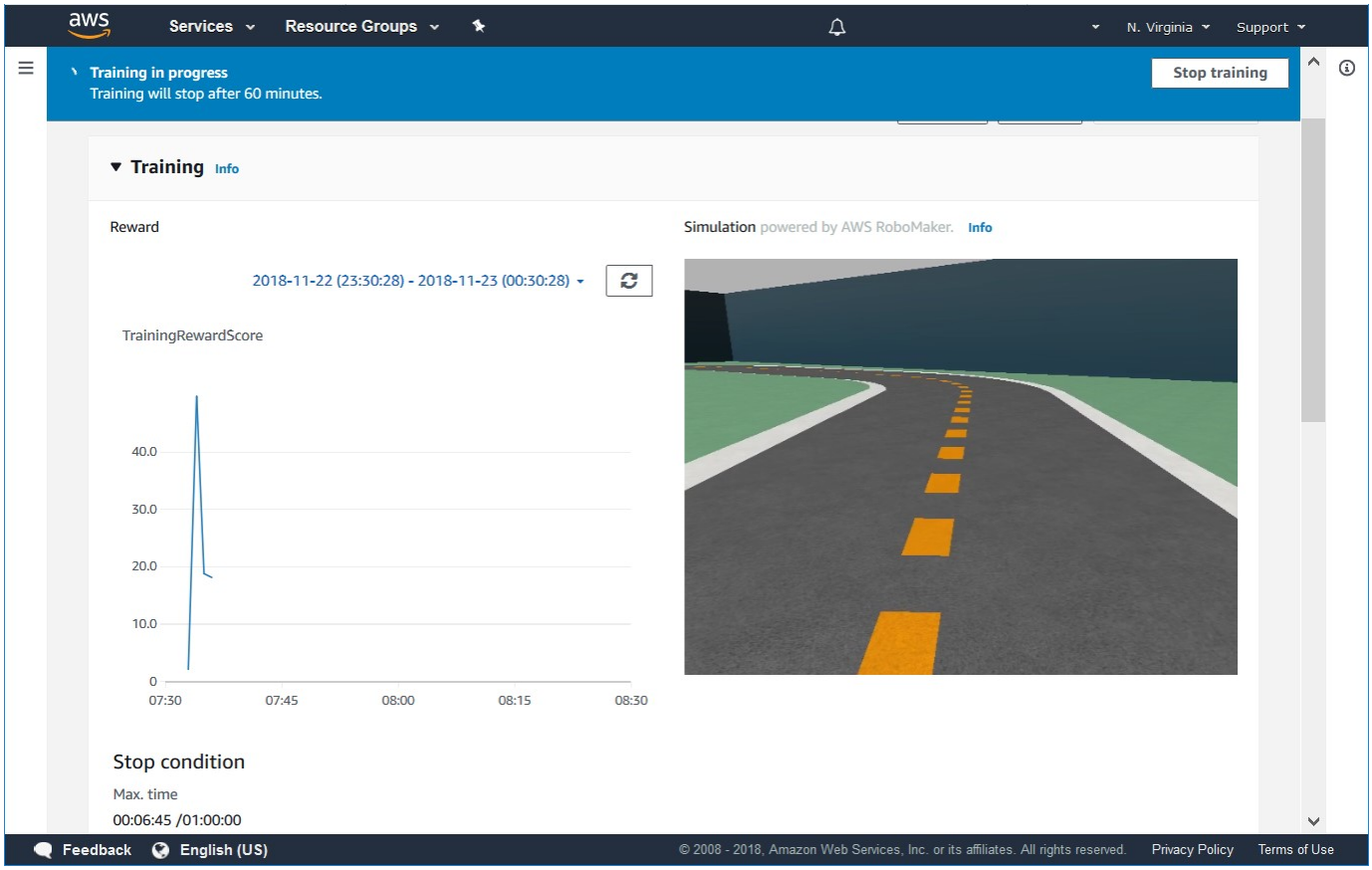
First, you have 10 minutes to choose a team name, do quick introductions, and come up with a reward function that your team thinks will get the car the furthest around the track. Tip: scroll up to the reward function section and look at the advanced reward function examples. What behavior do you want to incentivize to ensure your car stays on track and completes the largest part of the track? You will have to write your reward function using Python in the reward function section, and validate that it works, before you can start training.

At the end of your 10 minutes one nominated team member will start training the model using your reward function, and a stop condition of 15 minutes. Before you start training, please also note that it would be best to decrease the number of episodes between training to 5, and also decrease the number of epochs to 3. The last two conditions will help your model get more training iterations during the 15 minutes. If you can find a reward function sooner, you can always start training sooner, and just terminate when we say terminate.

Let's get going

Step 5: Model Training

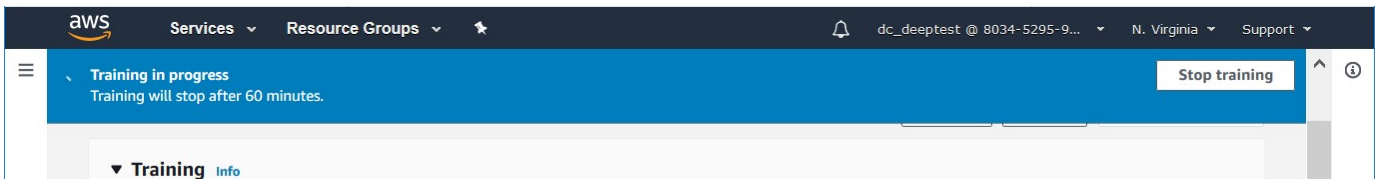
Training can take up to 6 minutes to initialize. We will cover the orchestration that happens after the first lab. Once your model has started training you should see the following on the Model details page, which can be found when you select your model from the Model list page.



The graph on the left is the TrainingRewardScore graph showing the cumulative reward your car receives for each episode. An episode is the number of steps from the car's starting point and the next termination state (completing the track or going off track). As your model trains you should see the reward line increase, however expect it to fluctuate initially. Once the cumulative reward does not improve much for each subsequent episode it is an indication that the model is not learning anything new and has converged.

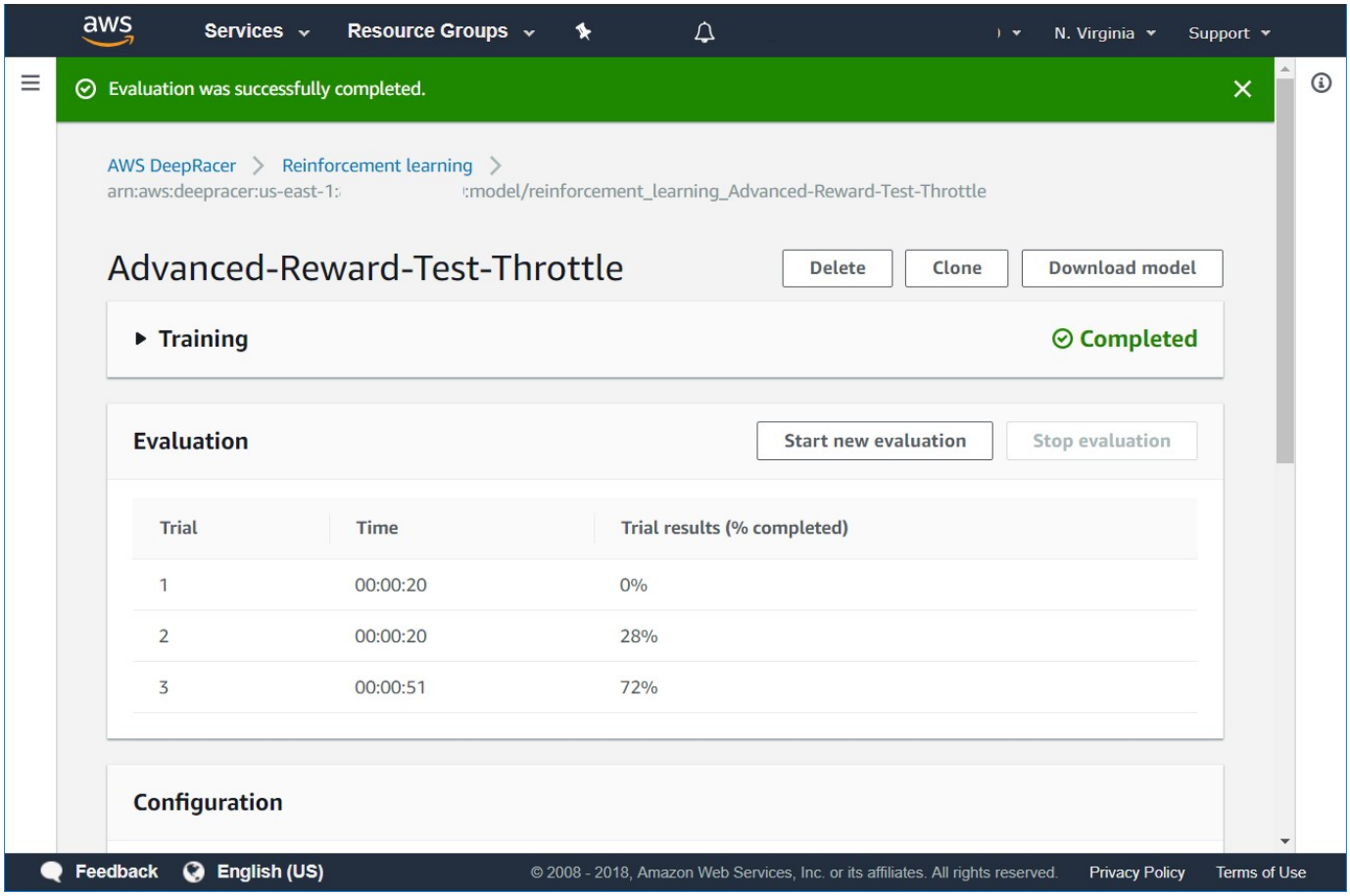
You can also visually inspect driving behavior by looking at the video stream from the simulator. You may notice that your car resets once it drives off the track. This is normal as the car is considered in a terminal state, and resetting it will restart a new episode. Furthermore, the car will reset at subsequent waypoints each time. This is done to help the car better explore the environment. Once you see your car successfully complete subsequent laps on the track, check to see if the reward graph is flattening out. This is a good indicator for you to stop training your model. In the event that your car is not completing the track and your model stops improving, as judged by the reward graph, consider stopping the training and proceed to tweak your model.

You can also inspect the Amazon SageMaker and AWS RoboMaker logs saved in Amazon CloudWatch. To access the logs you can select the 3 dots in the top right of your TrainingRewardScore graph, and select view logs or view in metrics. View logs will show you logs of the output from the Amazon SageMaker and AWS RoboMaker containers. They will contain more detailed information about your training job.



Step 6: Model Evaluation

Once your model has finished training, you will evaluate it in the simulator. This can be done from the model details page and choosing **Start evaluation** from the Evaluation section on the model details page. Here you can select the track on which you will evaluate your model,and also how many laps you want to evaluate it for. During evaluation your trained model will be used in the simulator to determine your lap time and lap progress.



In [ ]:

In [ ]:

In [ ]: