**Instructions**

Train a model in the [AWS DeepRacer console](https://console.aws.amazon.com/deepracer/home?region=us-east-1&utm_source=Udacity&utm_medium=Webpage&utm_campaign=Udacity%20AWS%20ML%20Foundations%20Course#getStarted) and interpret its reward graph.

**Part 1: Train a reinforcement learning model using the AWS DeepRacer console**

Practice the knowledge you've learned by training your first reinforcement learning model using the AWS DeepRacer console.

1. If this is your first time using AWS DeepRacer, choose **Get started** from the service landing page, or choose **Get started with reinforcement learning** from the main navigation pane.
2. On the **Get started with reinforcement learning** page, under **Step 2: Create a model and race**, choose **Create model.** Alternatively, on the AWS DeepRacer home page, choose **Your models** from the main navigation pane to open the **Your models** page. On the **Your models** page, choose **Create model**.
3. On the **Create model** page, under **Environment simulation**, choose a track as a virtual environment to train your AWS DeepRacer agent. Then, choose **Next.** For your first run, choose a track with a simple shape and smooth turns. In later iterations, you can choose more complex tracks to progressively improve your models. To train a model for a particular racing event, choose the track most similar to the event track.
4. On the **Create model** page, choose **Next**.
5. On the **Create Model** page, under **Race type**, choose a training type. For your first run, choose **Time trial**. The agent with the default sensor configuration with a single-lens camera is suitable for this type of racing without modifications.
6. On the **Create model** page, under **Training algorithm and hyperparameters**, choose the **Soft Actor Critic (SAC)** or **Proximal Policy Optimization (PPO)** algorithm. In the AWS DeepRacer console, SAC models must be trained in *continuous action spaces*. PPO models can be trained in either *continuous* or *discrete action spaces*.
7. On the **Create model** page, under **Training algorithm and hyperparameters**, use the default hyperparameter values as is. Later on, to improve training performance, expand the hyperparameters and experiment with modifying the default hyperparameter values.
8. On the **Create model** page, under **Agent**, choose **The Original DeepRacer** or **The Original DeepRacer (continuous action space)** for your first model. If you useSoft Actor Critic (SAC) as your training algorithm, we filter your cars so that you can conveniently choose from a selection of compatible continuous action space agents.
9. On the **Create model** page, choose **Next.**
10. On the **Create model** page, under **Reward function**, use the default reward function example as is for your first model. Later on, you can choose **Reward function** examples to select another example function and then choose **Use code** to accept the selected reward function.
11. On the **Create model** page, under **Stop conditions**, leave the default **Maximum time** value as is or set a new value to terminate the training job to help prevent long-running (and possible run-away) training jobs. When experimenting in the early phase of training, you should start with a small value for this parameter and then progressively train for longer amounts of time.
12. On the **Create model** page, choose **Create mode**l to start creating the model and provisioning the training job instance.
13. After the submission, watch your training job being initialized and then run. The initialization process takes about 6 minutes to change status from **Initializing** to **In progress**.
14. Watch the **Reward graph** and **Simulation video stream** to observe the progress of your training job. You can choose the refresh button next to **Reward graph** periodically to refresh the **Reward graph** until the training job is complete.

**Note:** The training job is running on the AWS Cloud, so you don't need to keep the AWS DeepRacer console open during training. However, you can come back to the console to check on your model at any point while the job is in progress.

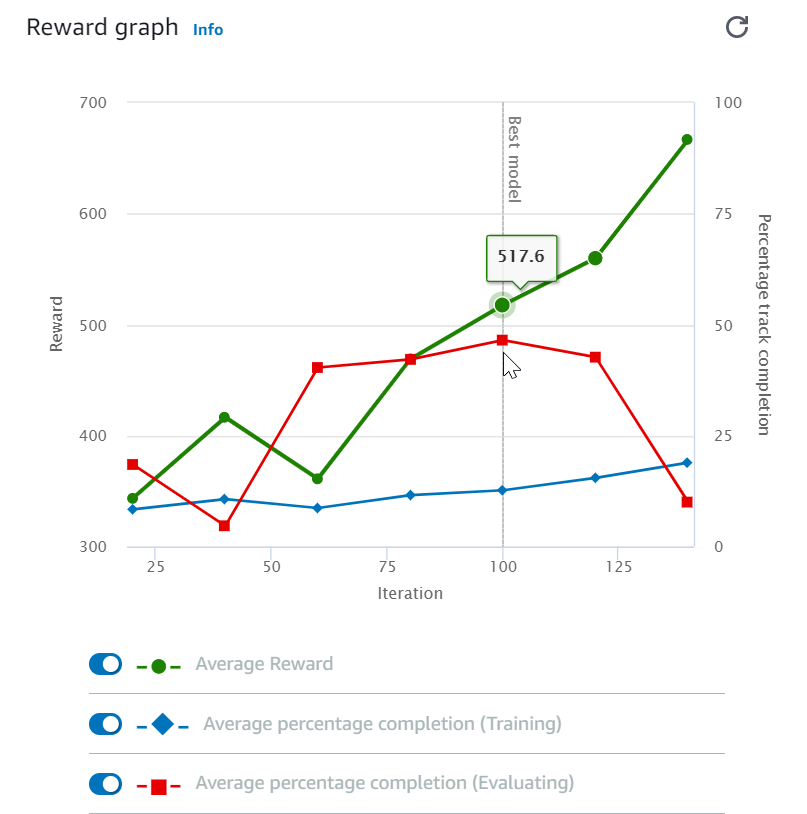
**Part 2: Inspect your reward graph to assess your training progress**

As you train and evaluate your first model, you'll want to get a sense of its quality. To do this, inspect your reward graph.

**Find the following on your reward graph:**

* Average reward
* Average percentage completion (training)
* Average percentage completion (evaluation)
* Best model line
* Reward primary y-axis
* Percentage track completion secondary y-axis
* Iteration x-axis

Review the solution to this exercise for ideas on how to interpret it.



As you train and evaluate your first model, you'll want to get a sense of its quality. To do this, inspect your reward graph.