Reinforcement Learning for Highway Driving

Marcelo García Escalante
mrgaresc@stanford.edu

Saba Bokredenghel
sababo@stanford.edu

May 5, 2023

1 Problem Statement and Task Definition

Autonomous driving is a rapidly growing field of research. The goal of this project is to develop a reinforcement learning agent that can drive a car on a highway. The agent will be trained in a simulated environment and will be evaluated on its ability to drive safely (with out collisions) and efficiently (at the highest speed allowed). The agent will be trained to drive in a highway with multiple lanes. The agent will be evaluated on its ability to drive safely and efficiently.

The environment that we will use is built on top of OpenAI Gym.[1] The environment is called "highway-v0"[2] We will use this environment to train and evaluate the agent using reinforcement learning algorithms we have learned in class such as Value Iteration or Q-learning, compare their outcomes and discuss the results.

2 Environment Description

2.1 Input (State Space)

The input to the agent will be the state of the environment. The observation space is a continuous space 5x5 2D-array: -inf, inf, (5, 5), float32

The state space captures the ego-vehicle's information plust the 4 closest vehicles in the highway, where the meaning of the features of the state are the following:

- **presence**: 1.0 if a vehicle is present, 0.0 otherwise.
- **x**: World offset of ego vehicle or offset to ego vehicle on the x axis.
- **y**: World offset of ego vehicle or offset to ego vehicle on the y axis.
- **vx**: Velocity on the x axis of vehicle.
- **vy**: Velocity on the y axis of vehicle.

Note: the coordinates are relative to the ego-vehicle, except for the ego-vehicle which stays absolute. The world frame is at the top left-corner of the highway. Examples of the state space are shown in appendix A

2.2 Output (Action Space)

The action space is a discrete space with 5 possible actions:

Action index	0	1	2	3	4
Action name	lane change left	idle	lane change right	accelerate	decelerate

3 Evaluation metric

The evaluation metric will be the average reward per episode. The reward function depends on 4 factors by default, but it can be changed.

- **Collision**: 0 if there is a collision, 1 otherwise.
- **Speed**: 0 to 1 depending on the speed of the ego-vehicle. 1 represents the maximum allowed speed
- **Close to right lane**: 0 to 1 depending on the distance to the right lane.
- **On road reward**: 0 or 1 depending on whether the ego-vehicle is on the road or not.

4 Baseline and Oracle

The baseline will be a random agent that selects an action uniformly at random. The oracle will be a human driver driving along the highway for 30 seconds as this is the maximum time allowed for the agent to drive in the environment per episode.

5 Related works

Our goal is to follow the implementation of Value Iteration and Deep Q-learning in[2] to solve the RL task stated in section 1. We might consider also other variants of Deep Q-Learning, like Double DQN, which is also stated in[2].
Other interesting work are included in [3] and [4], where DQN techniques become also the choice of approaches to these problems. In these research articles the question of safe decision making is discussed. There are two main difficulties: Collisions should never happen and the algorithms needs to take into account new observable states (e.g. unpredictable behaviour of other agents such as cars) in the environment.

6 Methodology

We will tackle this problem with Reinforcement Learning techniques. For this purpose, we will implement at least one model-based method (Value Iteration) and one model-free approach (Deep Q-Learning). Interesting in this sense will be the fact that we haven't covered Markov Decision Processes in continuous state spaces in the lectures. This will be the case for the Deep Q-Learning approach, which is based on function approximation.

7 Description of the challenges

The challenge of this project will basically be the following facts:

1. Formulating an MDP in a continuous state space.
2. Not all features from the state space might be useful, so we might have to come up with a feature extractor.
3. We need to combine Q-learning and Neural Network techniques and make ourselves more familiar with packages like pytorch or Tensorflow.
4. In our problem, we also have to handle two difficult parts. How can we ensure that collision will never happen? How to handle unpredictable behavior, like other agents with cars making their own decisions?

References

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.

[2] Edouard Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[3] Subramanya Nageshrao, Eric Tseng, and Dimitar Filev. Autonomous highway driving using deep reinforcement learning. *arXiv preprint arXiv:1904.00035*, 2019.

[4] Arash Mohammadhasani, Hamed Mehrivash, Alan Lynch, and Zhan Shu. Reinforcement learning based safe decision making for highway autonomous driving. *arXiv preprint arXiv:2105.06517*, 2021.

A Appendix: Raw state space samples

Sample 1

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.1200	1.0	0.0
vehicle 1	1.0	0.1900	0.0700	-0.3700	-0.1800
vehicle 2	1.0	0.4300	0.1100	-0.2000	0.1400
vehicle 3	1.0	0.6400	0.0000	-0.2000	0.0000
vehicle 4	1.0	0.8800	0.0800	-0.1200	0.0000

Sample 2

Table 1: Sample 2

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.0800	1.0	0.0
vehicle 1	1.0	0.2100	-0.0400	-0.1800	0.0000
vehicle 2	1.0	0.2400	-0.0800	-0.4800	0.0000
vehicle 3	1.0	0.5700	-0.0800	-0.3400	0.0000
vehicle 4	1.0	0.9500	-0.0800	-0.1600	0.0000

Sample 3

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.0500	0.6200	-0.1400
vehicle 1	1.0	0.0100	-0.0300	-0.0900	0.1400
vehicle 2	1.0	-0.2400	-0.0500	0.2200	0.1400
vehicle 3	1.0	0.2500	-0.0500	0.3500	0.1400
vehicle 4	1.0	0.8000	0.0300	0.3300	0.1400

Sample 4

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.0	1.0	0.0
vehicle 1	1.0	0.0600	0.0800	-0.4200	0.0000
vehicle 2	1.0	0.2500	0.0000	-0.4600	0.0000
vehicle 3	1.0	0.4800	0.0400	-0.5000	0.0000
vehicle 4	1.0	0.7700	0.0800	-0.4300	0.0000

Sample 5

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.0800	1.0	0.0
vehicle 1	1.0	0.1600	-0.0400	0.0700	0.0000
vehicle 2	1.0	0.2200	-0.0800	-0.2000	0.0000
vehicle 3	1.0	0.5700	-0.0800	-0.0600	0.0000
vehicle 4	1.0	0.9300	-0.0100	-0.0500	-0.3300

Sample 6

Vehicle	presence	x	y	vx	vy
ego-vehicle	1.0	1.0	0.0800	1.0	0.0
vehicle 1	1.0	-0.1600	-0.0800	-0.0300	0.0000
vehicle 2	1.0	0.2600	0.0000	-0.0200	0.0000
vehicle 3	1.0	-0.3300	-0.0400	-0.0600	-0.0100
vehicle 4	1.0	0.6800	-0.0800	0.0200	0.0000