Reinforcement Learning: Autonomous Race Car

Olivier Pham(mdopham@stanford.edu), Stephanie Sanchez(ssanche2@stanford.edu), Vishal Subbiah (svishal@stanford.edu)

Background and Motivation

Autonomous vehicles for the real world environment have been a long term ambition and in recent years endeavors towards this objective have shown very promising results. But what about autonomous driving for video games? If we can simulate autonomous driving in a game, it may give further insight to applying it in the real world.

Problem Statement

- **Reinforcement Learning** (RL) to train an agent in the Open AI gym Atari game, Enduro-v0.
- **Goal:** see if our agent could reach first place among 200 other cars by learning from its environment and actions, the intention of the game and develop an optimal policy to win.

Inputs and Outputs

- **Observation space**: 210*160 pixels 8-bit RGB image of the screen
- 9 **possible actions** which are: no operation, accelerate, right, left, brake, brake left, brake right, accelerate right, accelerate left.
- **Output**: new state s', Reward(s,a,s') = 200 - **Position** in the race at the state s'

The Model

Player Class:

- **MDP** class
- **Actions** are as described Inputs and Outputs.
- **Success and probability:** utilizes the Atari environment's step function to return a new observation based on the chosen action. Also returns a parameter that signals the end of the game which is our **end state**.

Reward Frame Extraction (position):

- a **quadruplets groups of pixels** is sufficient to distinguish the 10 different digits.

Reduced Action Space:

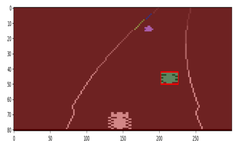
- Action space reduced to action 1 (accelerate), action 7 (accelerate right,) and action 8 (accelerate left).
- We ran each game for **1000 steps out of the 4500 steps** since the environment then changes every 1000 steps, thus changing the meaning of features.

Q-Learning:

. **Function approximation**: $\hat{Q}_{opt}(s,a) = w * \phi(s,a)$ where *s* is a raw pixel frame, *a* is an action, ϕ is the feature extractor , and *w is the weights updated by:*

$$w_{f_k} = w_{f_k} - \alpha * (Q(s,a) - (Reward(s,a) + \gamma * V_{opt}(s',a)) * f_v$$

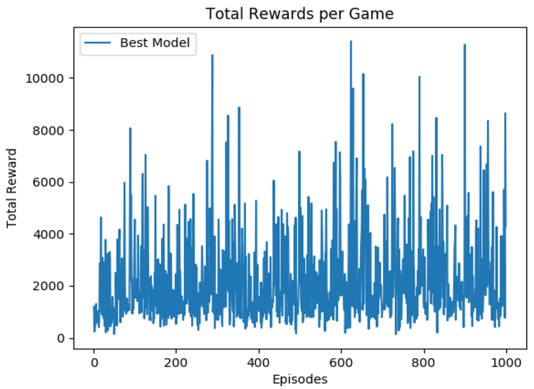
Feature Extraction: Every pair *(s,a)* was given a value equal to the number of cars on the road (including our agent). The detection of cars was done using image segmentation/labeling as seen in the picture).

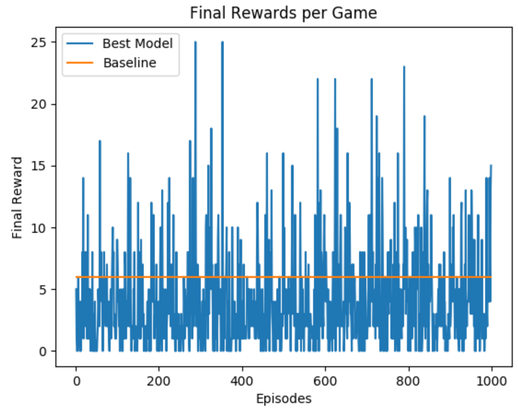


Results and Discussion

	Baseline	Value Iteration	Q-learning	Q-learning (Function Approx.)	Q-learning (Feature Extraction)
Result	194/200	199/200	187/200	187/200	175/200

We were able to show why Value Iteration would not do well for the given environment due to randomness in the number of frames per steps (unknown probabilities which this method relies on).
We implemented Q learning (function approximation) which gave an improvement in our performance as compared to our baseline. With the feature extraction we see a significant jump in our performance as compared to previous models.





Future Work

Our next steps would be to **expand the actions space** to all 9 moves from the current 3 we have shortlisted. We are currently running only the first 1000 iterations of the game to train the car in the same terrain. We would like to expand to work on **the different terrains**. From the limited games we have run we noticed a significant improvement with Q learning and so with enough episodes we should be able to win the race.

References

1. M. A. Farhan Khan, Oguz H. Elibol. *Car Racing using Reinforcement Learning*
2. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller. *Playing Atari with Deep Reinforcement Learning*
3. David Hershey, Rush Moody, Blake Wulfe. *Learning to Play Atari Games.*