# Solving OpenAI's Car Racing Environment with Deep Reinforcement Learning and Dropout

**Patrik Gerber***
University of Oxford
patrik.gerber@ccc.ox.ac.uk

**Jiajing Guan***
George Mason University
jiajingguan@gmail.com

**Elvis Nunez***
Brown University
elvis@brown.io

**Kaman Phamdo***
University of Maryland
kaman@phamdo.com

**Nicholas Malaya**
AMD Research
nicholas.malaya@amd.com

**Tonmoy Monsoor**
UCLA
mtonmoy@g.ucla.edu

## Abstract

Deep Reinforcement Learning methods have seen many successes in recent years, ranging from solving classical video games to beating world class Go players. However, successful models are typically trained for narrow, well-defined tasks using a vast amount of computation. They perform well in their defined task, but slight perturbations in the environment often cause disproportionate reductions in performance. In this paper, we provide an example which suggests that regularization methods could improve the generalizability of deep RL algorithms. We present a solution to OpenAI's Car Racing environment where the agent observes only a small subset of the state space during training by applying the DDQN-algorithm with dropout.

## 1 Introduction

Deep reinforcement learning methods have proven successful in solving well-defined computer games. In particular, the DDQN algorithm has achieved superhuman performance in several Atari games [4]. However, deep RL agents are limited by their sensitivity to perturbations in the environment. They can fail catastrophically when applied to environments that differ even slightly from where they were trained.

The OpenAI Gym's CarRacing-v0 environment [3] is a top-down view car racing game as shown in Figure 1. The goal is to visit the tiles of the racetrack as quickly as possible. The agent receives a reward of $-0.1$ at each time step and $\frac{1000}{N}$ for each track tile visited, where $N$ is the number of tiles in the racetrack. The state is represented by $96 \times 96$ RGB screenshots of the game. The game ends when the agent visits all tiles on the track or when 1000 frames have passed. This environment is considered solved by OpenAI's guidelines when the agent achieves an average score of 900 or above over 100 consecutive games. This game is particularly challenging because the racetrack is randomly generated, which introduces stochasticity and complexity into the environment. The challenge has been solved using a generative neural network model [1], but previous attempts using deep RL methods have not been successful [2].

In this paper, we describe a deep RL algorithm that uses a convolutional architecture with dropout that successfully solves the game. Remarkably, the model is trained on a limited environment made up of 3 tracks. This result shows that regularization methods such as dropout can mitigate the overfit usually exhibited by deep RL. To the best of our knowledge, this solution is the first to solve the challenge using deep RL.

Preprint. Work in progress.

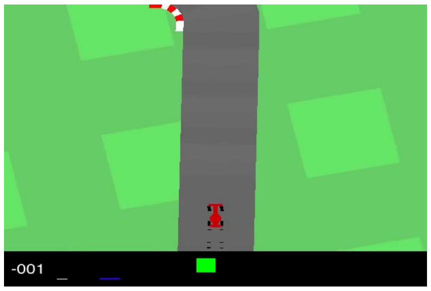

Figure 1: Screenshot of CarRacing-v0

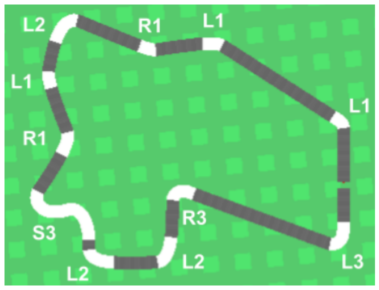

Figure 2: Characterization of different types of curves in the racetrack

## 2 Method

The car racing environment has a continuous action space with 3 values $(x, y, z)$ where $x \in [-1, 1]$ represents steering, $y \in [0, 1]$ represents acceleration, and $z \in [0, 1]$ corresponds to the brake. In our experiments, we discretized the state space into 5 actions: turn left $(-1, 0, 0)$, turn right $(1, 0, 0)$, accelerate $(0, 1, 0)$, brake $(0, 0, 0.8)$, and no operation $(0, 0, 0)$. These 5 actions were chosen based on the actions that a reasonable human player would use.

We adapted the DDQN algorithm [5] for our approach. The input to the Q-network is a $96 \times 96 \times 4$ image produced by stacking 4 consecutive frames of the game. As proposed in the original DQN algorithm [4], the architecture for the Q-network consists of 3 convolutional layers followed by 2 dense layers and an output for each of the 5 actions. Each hidden layer is followed by a rectified nonlinearity. Dropout is applied to the second convolutional layer. We train over a maximum of 3000 episodes and use early stopping to ensure that the best performing model is selected for analysis.

We trained several models with varying observability of the state space. First, we train the model on a single racetrack. Second, we randomly alternate between 3 fixed racetracks. Third, the agent is shown random racetracks from the state space. All models are tested on random racetracks. We performed a baseline experiment with no dropout and experimented with the dropout probability. The results from the best performing models are displayed in Table 1.

## 3 Performance Analysis

| **Environment** | **Dropout** | **Average Score** |
|---|---|---|
| 1 Fixed Track | None | $849.99 \pm 78.72$ |
| 3 Fixed Tracks | None | $853.88 \pm 127.71$ |
| Random Tracks | None | $854.83 \pm 107.14$ |
| 1 Fixed Track | 0.7 | $894.38 \pm 24.5$ |
| 3 Fixed Tracks | 0.5 | $906.67 \pm 23.6$ |
| Random Tracks | 0.5 | $892.62 \pm 41.48$ |

Table 1: Average scores over 100 consecutive games on random racetracks

As shown in Table 1, the average scores over 100 random games improved in all three environments when dropout was applied to the Q-network. This suggests that dropout acts as an effective regularizer by improving the agent's performance in situations it has not observed during training. In order to investigate the generalizability of these models, we developed a curve classification algorithm. Each curve in the racetrack is charactertized as either a Left, Right, or S-shaped curve. Then, the steepness of the curve is ranked from 1 to 3, where 1 represents a shallow curve and 3 represents a steep curve. For each of these different types of curves, we record the percentage of tiles cleared.

Figure 3 compares the performance of the models trained on a single racetrack. Even though both models were trained on the same racetrack, the model with dropout performed better than the model without dropout, especially on curves that agent did not see examples of during training. This indicates that using dropout is an effective regularizer and improves the generalizability of these models.

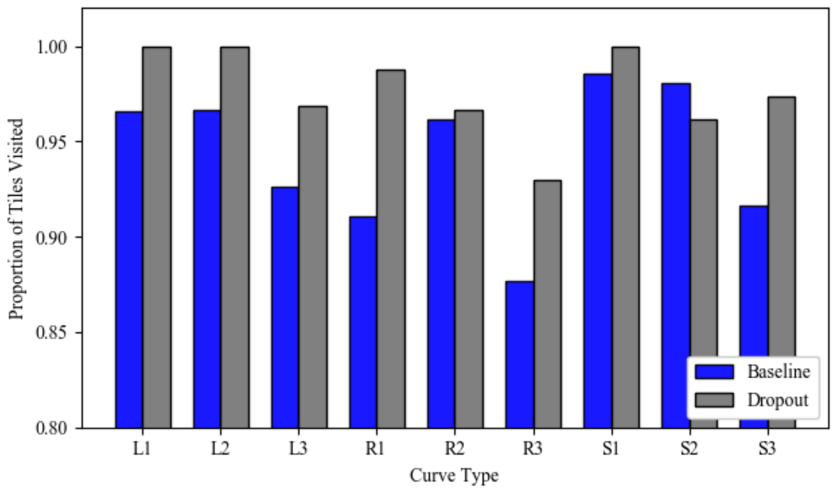

Figure 3: Curve performance for models trained on one racetrack with and without dropout

## 4 Conclusion

We demonstrate a deep RL solution to CarRacing-v0 where the agent has limited access to the state space. Even with partial knowledge about the environment, the agent is capable of making informed decisions. By applying dropout as a regularization technique, we improve the algorithm's ability to generalize to situations it has not observed during training. These results suggest that regularization techniques could be used to make deep RL algorithms more robust under limited observations of the environment.

## References

[1] David Ha and Jürgen Schmidhuber. World models. *CoRR*, abs/1803.10122, 2018.

[2] M. A. Farhan Khan and Oguz H. Elibol. Car racing using reinforcement learning. 2016.

[3] Oleg Klimov. CarRacing-v0. `https://gym.openai.com/envs/CarRacing-v0/`. Accessed: 2018-09-01.

[4] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[5] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.