Examining an iterative development of reward functions to generate autonomous driving models using reinforcement learning

Sashrika Pandey
Secondary Student Training Program
The University of Iowa

https://cs.uiowa.edu/people/denise-szecsei-0

Denise Szecsei, Associate Professor, Ph.D.

Introduction/Background

Machine learning involves the development of a predictive model that is initially trained on several data sets before being exposed to a test data set from which it generates a projected result. Reinforcement learning is a subset of machine learning that trains a model by using a reward function to encourage certain behaviors. After a model completes an iteration of training, it is rewarded for adhering to the criteria outlined by the program, which consequently results in future versions of the model also valuing such conditions (Qiang & Zhongli, 2011, p. 1143).

As reinforcement learning involves a trial and error process, it is often lengthy and time-consuming, which has led to the practice of pre-training models before exposing them to the rest of a training data set. Past studies have used neural networks to process this data so that the model does not have to overcome an initial learning curve (Kim, Cha, Ryu, & Jo, 2019, p. 2). The use of past data and trends is practical when a system based on reinforcement learning must process and filter massive amounts of signals (Moon, Cheong, Yeom, & Woo, 2019, p. 345).

Research Objectives

Through the use of iterative models, this study aimed to identify the combinations of parameters that would yield a virtual car model that could complete a lap around a track with the most consistency and the fastest time. In particular, the research centered around examining the relationship between training clones of several models for varying amounts of time and identifying how past iterations of a model could impact the performance of one that is retrained with a new reward function. The study considered the virtual car's actions at certain parts of the track across various trials to assess the performance of the model in different track segments.

Method

Amazon Web Services hosts the DeepRacer virtual environment in which models are trained and then evaluated in timed trials. For this study, the various iterations of models were trained on the re:Invent track, which includes both straight and curved sections. After a training period of a user-determined time, the model was evaluated for five testing laps, and the percent of track completed along with the time per trial were stored in resulting logs.

Base models processed data directly from the virtual car, including its position on the track, speed, and steering. Characteristics of the track included waypoints, which are a set of predetermined coordinates on the track, and the width of the track. The next stage of the process was to clone a previous model, which would retain its experience from prior training, and modify its reward function or to train a model based on a new reward function. The performance of cloned models could be influenced by the training of previous iterations. The car's laps during the evaluation period were extracted from generated logs and the path of the car was charted. The results indicated how a change in the reward function impacted the car's behavior. Parameters included in the reward function were reevaluated based on how the car would adjust

its decisions to maximize its rewards. The consistency of the car in completing evaluation laps and its trial times were used to evaluate if a model could be cloned for future development.

Results

Different formulas for assigning rewards were used for straight versus curved track segments. The methods used to determine curvature were: finding the relative distances between waypoints; the use of Bezier curves to determine parametric functions that connected waypoints and computing their curvature; and a calculation that found the cosine of the angle formed by the vectors connecting consecutive waypoints. The model generated using vectors and angles had the fastest completed evaluation lap time of 19.761 seconds in comparison to the 19.967 seconds of the Bezier curve model and the 20.219 seconds of the relative distance model. The fastest lap time for the initial model, which did not distinguish between curvatures, was 25.655 seconds.

Model development benefitted from rewarding increased speed and efficiency in completing a lap, as indicated by the number of decisions, or steps, that the car made. When rewards were given proportional to the cube of the speed and the progress relative to steps completed, the model's performance improved. The series of models based on the ninth cloned model of the first base model indicated that a combination of both types of rewards in relation to curvature positively impacted performance. Each of the three methods used to distinguish between straight and curved segments yielded lower lap times than those of the series of cloned models. This was further corroborated by the model series that was cloned from the first reward function. Minor adjustments were made in successive iterations but performance either stayed constant or degraded, which is likely due to the increase in conditions in the reward functions. These constraints would cause the model to narrow its focus while training and might have led to contradictions between the new reward function and prior training experience.

Conclusions/Implications

The model's success was largely reliant on the accuracy of the method used to predict the curvature and the rewards that were available depending on whether the track was determined to be straight or curved. Calculating curvature with the vector and angle method and rewarding increased speed and progress led to improvements in the model's timed trials. Models developed as newly initialized reward functions performed better than the corresponding clones, indicating that iterative models were not as effective. The aforementioned results could be used to optimize both a virtual and a physical car's path by training models that consider a variety of parameters. Reinforcement learning could be used in navigation to train systems based on sensor input since reward systems could accelerate the processing rate of massive datasets. Moreover, reward functions could be used to identify parameters that significantly impact the model's behavior.

References

Kim, J., Cha, S., Ryu, M. & Jo, M. (2019). Pre-training framework for improving learning speed of reinforcement learning based autonomous vehicles. *2019 International Conference on Electronics, Information, and Communication (ICEIC)*, Auckland, New Zealand, 1-2. doi: 10.23919/ELINFOCOM.2019.8706441.

Moon, J., Cheong, M., Yeom, I. & Woo, H. (2019). Deep reinforcement learning based sensor data management for vehicles. *2019 International Conference on Information Networking (ICOIN)*, Kuala Lumpur, Malaysia, 345-349. doi: 10.1109/ ICOIN.2019.8718108.

Qiang, W. & Zhongli, Z. (2011). Reinforcement learning model, algorithms and its application. *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*, Jilin, China, 1143-1146. doi: 10.1109/MEC.2011.6025669.