Deep reinforcement learning approach for autonomous driving

Y.O. Anisimov, D.A. Katcai
South Ural State University, Chelyabinsk, Russia
e-mail: yan@yanchick.org

Abstract. The paper describes a control system for an autonomous driving car. Control systems are based on deep neural networks. Due to the impossibility of preliminary formation of a high-quality training sample for problems, the use of reinforcement learning is considered. Possible variants of the modeling environment implementation are described.

Introduction.
Autonomous driving is an active area of research with dramatic progress in recent years and could play a major role in the future of transportation systems, due to increased safety and efficiency. Advanced control systems with a certain degree of autonomy are already available in consumer cars. A key challenge in this area has been the difficulty in incorporating the ability to handle rare unseen situations as they follow a very long-tailed distribution. Alternatively, learning the whole autonomous driving stack in an end-to-end fashion offers a potentially more scalable way of achieving better autonomy. Recent works have successfully used imitation learning to train agents by imitating the actions of a human driver. While this approach has proven effective and easy to train, it requires a significant amount of data to train and does not generalize well to out-of-distribution situations. A modern car with an autonomous driving module is a rather complex device. It consists of several components [1].

Formulation of the problem.
A typical autonomous driving stack consists of different modules for perception and control. The driving stack also consists of several handcrafted logical rules to separately deal with different situations that it might encounter. The main modules are:

- Computing device. Usually, it is a fairly powerful computer with network access. The main task is generating a control signal based on information from all sensors.
- Satellite navigation systems. This subsystem also has a high-precision digital road map. This subsystem provides information about the object position on the trajectory.
- Lidars for providing information about the distance of the vehicle relative to surrounding objects.
- Vision camera. It is used for obtaining information about the real-time situation on the road.
- Executive device. By design, this device can be either a separate electronic unit that generates a signal into the on-board computer of a car, and mechanical - imitating real human actions.
All sensors can be divided into two classes: those that produce information suitable for analysis (lidar and satellite navigation system) and non-formalized information (vision system).

Recently, neural networks have been the most effective for video and image analysis. State-of-art network architectures have appeared that solve various problems: recognition of images in a picture, tracking an object in a video stream, analyzing a video stream to search for anomalies [2] [3]. The main feature of neural networks is that they can receive various combinations of data - a video stream together with information from lidars and satellite systems, and immediately generate a control signal [4].

Solving this class of problems requires a high-quality training sample. Training sample is an array of samples containing input-output pairs. This approach is not good for autonomous driving. It is impossible to create a significant array of all possible situations on the road and the correct solution to it. This is especially complicated by the presence of dynamics during the car movement. In this regard, the reinforcement learning scheme is proposed [4] [5].

**Training scheme.**

Reinforcement learning consists in training the machine learning models to make a sequence of decisions. The agent learns to achieve a goal in an uncertain, potentially complex environment. In reinforcement learning, the artificial intelligence faces a game-like situation. The computer employs trial and error to come up with a solution to the problem. To force the machine to do what the programmer wants, the artificial intelligence gets either rewards or penalties for its actions. Its goal is to maximize the total reward.
Despite the fact that he sets the reward policy, that is, the rules of the game, he gives the model no hints or suggestions on how to solve the game. It’s the model who must figure out how to perform the task to maximize the reward, starting from totally random trials and finishing with sophisticated tactics and superhuman skills. Leveraging search capabilities and multiple trials, reinforcement learning is currently the most effective way to demonstrate the creativity of a machine. In contrast to human beings, artificial intelligence can gather experience from thousands of parallel gameplays if a reinforcement learning algorithm is run on a sufficiently powerful computer infrastructure.

In math we have a model:

$$S_t = \text{Env}(S_{t-1}, u_t),$$

where $\text{Env}(\ )$ - model of environment, $u_t$ - control signal.

Environment gets a reward:

$$R_t = \text{R}(S_t)$$

$R$ - model of reward.

Reward is a negative discrete value indicating the agent errors: the higher the value in absolute value, the more dangerous error is made by the agent. The value of the value itself is assigned expertly: for example, collision -100, overspeeding -10, and so on.

The goal of training a neural network is to maximize the reward following a sufficiently large number of iterations:

$$\text{argmax} = \sum y_t R(\text{Env}(S_{t-1}, \text{NN}(R_t, y_t, u_t))),$$

where $y_t$ - discount, $\text{NN}(\ )$ - model of neural network(agent), $y_t$ - observable state.

In this case, the modification of the neural network weights occurs at the end of each iteration. Implementation algorithm is as follows:

Initial neural network $\text{NN}(\theta, S)$

for Episodes not end do

for $t = 0..30$ sec do

Get reward $R_{t-1}$ and observable state $y_t$

Neural network generate control signal $u_t = \text{NN}(\theta, y_t, R_{t-1})$

Append array for learning neural network $\{y_{t-1}, y_t, R_{t-1}, u_t\} u_t$

Tuning neural network

end

Append to array reward for logging

end

Implementation of the environment.

The main challenge of reinforcement learning is preparing the simulation environment. When the model has to go superhuman in Chess, Go or Atari games, preparing the simulation environment is relatively simple. When it comes to creating a model capable of driving an autonomous car, creating a realistic simulator is crucial before letting the car drive down the street. The model has to figure out how to brake or avoid a collision in a safe environment, where even a thousand cars would be sacrificed at the minimal cost. Transferring the model out of the training environment into the real world is where things get tricky.[6]
In this work we use CARLA (Car Learning to Act) – an open simulator for urban driving. CARLA has been developed from the ground up to support training, prototyping, and validation of autonomous driving models, including both perception and control. CARLA is an open platform. Uniquely, the content of urban environments provided with CARLA is also free. The content was created from scratch by a dedicated team of digital artists employed for this purpose. It includes urban layouts, a multitude of vehicle models, buildings, pedestrians, street signs, etc. The simulation platform supports flexible setup of sensor suites and provides signals that can be used to train driving strategies, such as GPS coordinates, speed, acceleration, and detailed data on collisions and other infractions. A wide range of environmental conditions can be specified, including weather and time of day.[7]

Implementation of the agent.
An agent is based on convolution neural network. It consists of a cascade of three convolutional layers, in pair with a pooling layer, and a fully connected layer at output. Network training takes 20 epochs per episode. The 500 episodes are enough to train an agent.

Conclusion
Using reinforcement learning, you can train a neural network that will control a moving object using the entire dataset - video stream, lidars, and coordinates from a satellite navigation system. A trained network through open standards can be tested on full-scale models without fear of situations that were not foreseen at the design stage. At the same time, the costs of forming and marking up the training sample will be minimal.

References
[1] Campbell S. et al., "Sensor Technology in Autonomous Vehicles: A review," 2018 29th Irish Signals and Systems Conference (ISSC), Belfast, 2018, pp. 1-4.
[2] Krizhevsky A., Sutskever I., Hinton GE Imagenet classification with deep convolutional neural networks // Advances in neural information processing systems. - 2012. - With . 1097-1105.
[3] LeCun, Y., Kavukcuoglu, K., Farabet, C. Convolutional networks and applications in vision. In Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS) (2010). IEEE 253-256
[4] Xia W., Li H. and Li B., A Control Strategy of Autonomous Vehicles Based on Deep Reinforcement Learning, " 2016 9th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, 2016, pp. 198-201, doi: 10.1109 / ISCID.2016.2054.
[5] Rao Q. and Frtunikj, J. Deep Learning for autonomous driving Cars: Chances and Challenges, 2018 IEEE / ACM 1st International Workshop on Software Engineering for AI in Autonomous Systems (SEFAIAS) , Gothe nburg, 2018, pp. 35-38.
[6] Brockman G. et al. OpenAI Gym [Electronic resource] // arXiv.org. 2016. Date of update: 5.06.2016. The URL: https://arxiv.org/abs/1606.01540 (reference date: 29.08.20 20).
[7] Dosovitskiy, A., Ros, G., Codevilla, F., López, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. URL: https://arxiv.org/abs/1606.01540 (reference date: 29.08.20 20).