Deep Reinforcement Learning for Street Following in Self-Driving Cars

Shahd Safarani

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

03. December 2018

Outline

What are Self-Driving Cars?

- Robotic systems are able to drive and navigate fully autonomously, relying - just like humans - on a comprehensive understanding of the immediate environment while following simple higher level directions (e.g. turn-by-turn navigation commands).

Source: [1]

About Self-Driving Cars

- Researchers and AI experts predict to have car robots ready-to-use in one or two decades (e.g. Rodney brooks prediction in "My Dated Predictions").

Rod. Brooks, source: [2]

About Self-Driving Cars

Utopian View

- Save lives (1.3 million die every year on the world's roads due to car accidents more than 90% of which caused by human error)
- Eliminate car ownership
- Increase mobility and access
- Save money (e.g. for damages caused by accidents)
- Make transportation efficient and reliable.

Dystopian View

- Eliminate jobs in the transportation sector
- Ethical Issues (e.g. society etc.)
- Security

Autonomous Driving Agent

An autonomous driving agent should be able to:

- Recognize its environment (lane detection, traffic sign recognition etc.)
- Keep track of the environment's state over time (self-localization, the occlusion of objects)
- Planning its actions based on its observations

A Car Robot, source: [3]

Recognition

- Recognition of the static environment.
- Identifying entities in the surrounding environment.
- Examples of this are pedestrian detection, traffic sign recognition, etc.
- It includes detection and recognition tasks of static objects (Mostly vision-based tasks).

Traditional methods relied on two stages:

- Handcrafting features by low-level Feature extraction (SIFT, HOG and Haar-like).
- Classification using shallow trainable architectures (e.g. SVM classifiers).

Recognition

DNNs/CNNs dominated since AlexNet in all computer vision tasks due to:

- Having deeper architectures and learning more complex features.
- Learning the features relevant to the task rather than designing features manually.
- Its expressivity and robust training to generalize and learn informative object representations.

Prediction

- Information integration over time is mandatory, since the true state is revealed as you move.
- Examples of this are localization and mapping, ego-motion, the occlusion of objects, etc.
- Learning the dynamics of the environment (Being able to predict future states and actions).
- It includes tracking tasks (object tracking).
- Mainly, many features are extracted and then tracked over time.

Traditional methods for localization and mapping has a standard pipeline including:

- Low-level Feature extraction (e.g. SIFT).
- Information integration by tracking extracted features (e.g. KLT tracker).

Prediction

DeepVO for localization:

- End-to-end learning model for Visual Odometry, using RCNNs.
- Achieved competetive results, compared to the state-of-the-art methods used for localization and mapping.

DL Preferable to traditional approaches because:

- They need to be carefully designed and specifically fine-tuned to work well in different environments.
- Some prior knowledge required.
- RNNs are able to memorize long-term dependencies and tackle POMDPs (Partially Observable MDPs), while traditional methods (e.g. Bayesian Filter) based on Markov Assumption.

Planning

- Movement Planning to move around and navigate.
- **Traditionally** formulating the control problem as an optimization task.
- Many assumptions have to be made to optimize an objective.
- **Reinforcement learning** seems to be promising for planning and control aspects.
- Especially, when handling very complex environments and unexpected scenarios.

- **Standard Approach:** Decoupling the system into many specific independently engineered components, such as perception, state estimation, mapping, planning and control.

- **Drawbacks:**
  - The sub-problems may be more difficult than autonomous driving (e.g. Human drivers don't detect all visible objects while driving).
  - Sub-tasks are tackled and tuned individually, which makes it hard to scale to more difficult driving scenarios due to complex inter-dependencies.
  - As a result, they may not combine coherently to achieve the goal of driving.

Autonomous Driving and DeepRL

- **An alternative:** a combination of Deep Learning and Reinforcement Learning (DeepRL) to tackle the autonomous driving task end-to-end [4].
- RCNNs responsible for recognition and prediction (representation learning), while RL responsible for the planning part.
- RNNs are required due to some scenarios that include partially observable states in autonomous driving.
- Learning relevant features for the driving task accomplished by reinforcement learning with a reward signal corresponding to good driving.

Reinforcement Learning

- **Reinforcement learning** is a general-purpose framework for decision-making.
- An agent operates in an **environment** and can act to influence the state of the environment.
- The agent receives a **reward signal** from the environment after taking an action.
- **Success** is measured by a reward signal.
- The agent learns good and bad actions, aiming in the long run to select actions that **maximize** the expected reward.

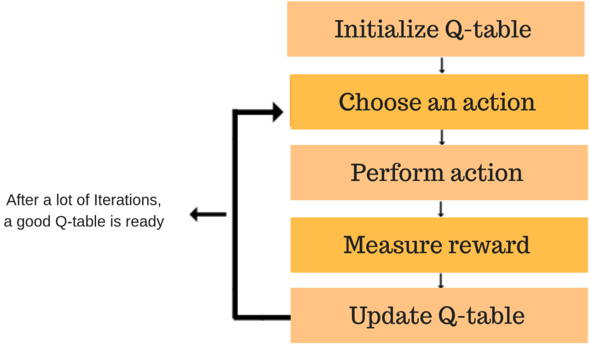
RL terms:

- The model developed under the **Markov Decision Process (MDP)** framework (State Space, Action Space, Reward Function and State Transition Probabilities).
- **Policy**: agent's behavior function.
- **Value function**: how good is each state and/or action (e.g. state-action value function: Q(s,a) represents the expected return when being in a state s and following the policy p till the end of the episode.
- **The goal**: finding a policy that maximizes the total rewards from the source to the terminal states.

Q-Learning

- Q-learning is one of the commonly used algorithms to solve the MDP problem.
- It is an iterative algorithm to get as much information as possible when exploring the world.
- Use **any policy** to estimate Q that maximizes future rewards.
- The Q-learning algorithm is based on the Bellman equation.
- Exploration/Exploitation dilemma needs to be solved carefully.

Q-Learning



Q-Learning Algorithm, source: [5]

Q-Learning

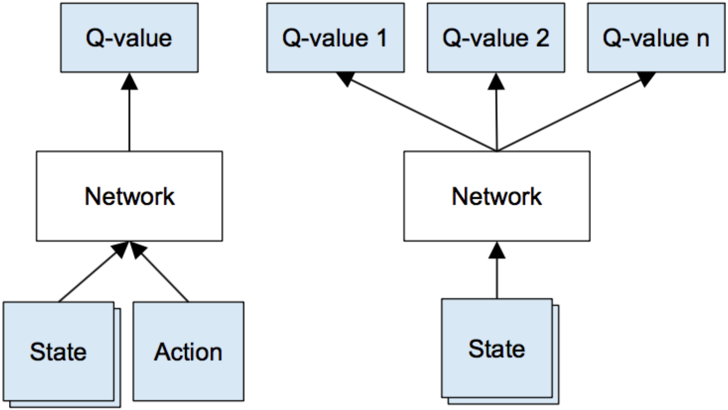
$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

Bellman Equation, source: [6]

Deep Q-Networks (DQNs)

- When the state space is very large (**continuous**), the q-function can't be formulated as a table.
- **An idea:** Formulating q-function as a parameterized function of the states and actions Q(s, a, w).
- Try to approximate the q-function using DNNs.
- Fitting the parameter w to the task becomes the objective.

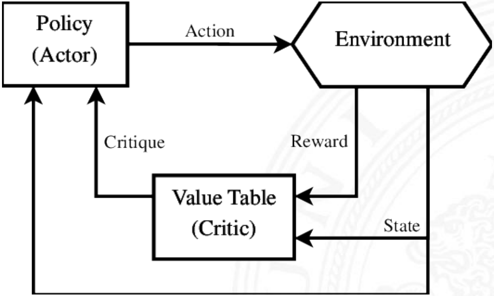
Deep Q-Networks (DQNs)



DQN, source: [7]

DDPG for Continuous Actions

- DQNs are modified for continuous action spaces.
- An example method is **DDPG** involves two networks: The *Actor* and the *Critic*.
- For more details on DDPG, please see [8].



Actor Critic Model, source: [9]

Deep Recurrent Q Networks (DRQN)

- Q-learning algorithms based on the Markov assumption, which is not valid in partially observable scenarios.
- POMDPs are tackled using information integration over time.
- The recurrent neural networks (RNN) present themselves as a natural framework to tackle POMDPs.
- In the literature, RNNs proved to be performing well when being integrated in DQNs.

About the Paper

Title: Learning to Drive in a Day [10]

Submission Date: 11-Sep-2018

Source: *Wayve* is pioneering artificial intelligence software for self-driving cars in UK.

About the Paper

- The first application of deep reinforcement learning on-board a self-driving car.
- Just like humans when learning how to ride a bicycle (safe environment provided + high-level control).
- No rules to be followed for following the lane.
- No maps of the environment provided implicitly (e.g. lane borders in the reward signal).
- The system able to learn to lane follow from scratch without knowledge of lane position under thirty minutes of training – all done on-vehicle.
- Environment perception using a single monocular forward-facing video camera.

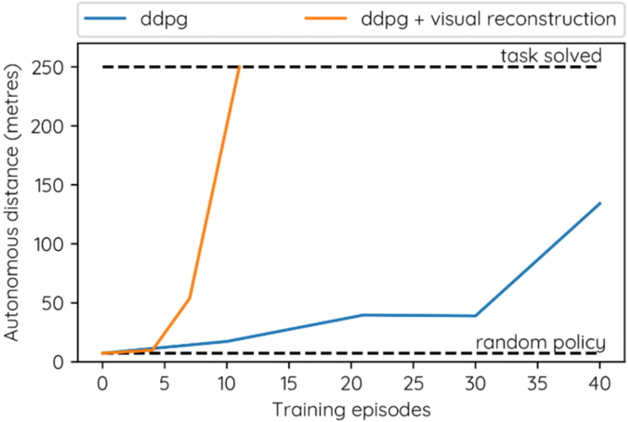
Problem Formulation and Methods

- State space: a monocular image as input, together with the observed vehicle speed and steering angle (No need for RNNs).
- State representation by two approaches: Feeding the image through CNNs or using small compressed representations as input.
- Two-dimensional action space: steering angle in the range [-1, 1] and speed setpoint in km/h.
- Reward: the distance travelled by the vehicle without the safety driver taking control.
- DDPG is used for planning.

Experimental Setups

- Learning is performed online with no dense 3D maps and hand-written rules.
- All computation is done on-board using a single NVIDIA Drive PX2 computer.
- Trained in simulation for figuring out the best hyperparameters.
- For real-world driving, a 250 meter section of road is used.
- Testing two approached for state representation.

Results



A comparison between ddpg with and without Variational Autoencoders, source: [10]

Discussion

- The agent is able to learn street-following online in 30 minutes on-vehicle.
- With RL advancements, this framework can generalize and scale to more complex scenarios.
- "The method here solved a simple driving task in half an hour – what more could be done in a day?" [10].

- The reward signal is general. Designing a more effective function is future work considering ethical issues and safety.
- State representation could be improved drastically (e.g. RNNs usage).

Conclusion

- Deep Learning + RL is promising and can be leveraged in Autonomous Driving.
- Choosing the reward function is challenging.
- Simulation can be used for tuning before training in the real world.
- Training in real-world is much more complex than in a simulated world.
- On-board a car robot, RL agents could be embedded and efficiently trained in complex real scenarios.

Questions

Thank You!
Any questions?

[1] https://inhabitat.com/100-self-driving-cars-set-to-hit-swedens-public-roads-in-2017/.

[2] https://people.csail.mit.edu/brooks/.

[3] https://medium.com/@george.seif94/the-future-of-self-driving-cars-2c06d988e996.

[4] A. Sallab, M. Abdou, E. Perot, and S. Yogamani. "Deep Reinforcement Learning framework for Autonomous Driving". In: *Electronic Imaging* 2017 (Jan. 2017), pp. 70–76. DOI: 10.2352/ISSN.2470-1173.2017.19.AVM-023.

[5] https://medium.freecodecamp.org/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc.

[6] https://stackoverflow.com/questions/40121969/q-learning-updating-frequency.

[7] https://ai.intel.com/demystifying-deep-reinforcement-learning/.

[8] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. "Continuous control with deep reinforcement learning". In: *CoRR* abs/1509.02971 (2015). arXiv: 1509.02971. URL: http://arxiv.org/abs/1509.02971.

[9] https://www.researchgate.net/figure/Structure-of-the-actor-critic-learning-methods_fig1_293815876.

[10] https://arxiv.org/abs/1807.00412.