Self-driving cars learn by imagination

Sara Mahmoud[1], Henrik Svensson[2]

[1,2]: *Interaction Lab, School of Informatics, University of Skövde*

Sara.mahmoud@his.se

Introduction

This short paper outlines a cognitively inspired "imagination" based approach for interactive machine learning for self-driving cars. The approach is based on learning from experience but with the idea of optimizing the driving from "dreaming" or "imagining" driving on different roads and is part of the general approach developed in the EU Horizon 2020 project Dreams4Cars (Lio et al., 2017). The work presented here builds, partly, on the general idea of cognition as simulation of perception and action (e.g. Hesslow, 2002) and robot models based on that idea (summarized in Svensson, 2013). While the quality of the imagined worlds for cars and humans are quite different, the model presented here takes inspiration from a possible facilitating role of dreams for learning (Svensson, Thill, & Ziemke, 2013). The approach is thought to increase the agent's performance even by learning from internally generated scenarios not seen by the agent. At the heart of the approach to improving learning in the driving domain is what we call an episodic generator simulation that can generate different driving scenarios to train the agent to drive and learn without manually programming every scenario. During the learning phase in a certain road scenario, the agent learns to behave in these situations but may not perform well in new situations. The episodic generator generates suitable roads that improves the learning performance, for example, by recreating variations of what the agent performed poorly on.

Self-driving cars are interactive agents that need to learn from experience. Reinforcement learning allows the agent to explore the environment, perform actions and learn from these actions. This approach has been used for discrete state environments. However, it has started to flourish in continuous state space as well. Lillicrap et al. (2015) designed a reinforcement model with deep learning that can learn different games with the same configurations and hyper-parameters with continuous control that they patented in 2017. Their model is a deep network of convolution layers followed by filter layers and the output layer. In this work, multiple openAI environments[1] were used. OpenAI environments are designed to handle the reinforcement learning needs, such as the state space that the agent observe in each step, the accepted actions that agent can take and the rewards obtained from each action in each state (Brockman et al., 2016). This work shows promising opportunities for deep reinforcement learning that can learn to play video games (Singh, Okun, & Jackson, 2017), however this work is still restricted by the game world image pixels as input. If we want to deploy dreaming learning to the real world, this representation of game image pixel inputs doesn't fit for real world deployments because of the major differences of the agent's view of the world. In order to deploy such models for real world agents, the agent needs to get meaningful sensory inputs that can then to some extent match the real world, such as distance readings from objects.

However, imaginations in an off-line world may be useful. For example, Mahedevan (2018) argues that what he calls "imagination science" is the future of AI. Imagination science is according to him different from data science. Data science is based on statistical calculations for finding correlations in the given data, such that machine learning of image and object recognition doesn't go much beyond labeling items, while imagination science strives to construct machines that not only learns to categorize and label the world but also create novel worlds. In imagination science, the system is able to reason and find causal explanations.

Imagination is important in several human tasks and a possible advantage of such an ability is to come up with a new usage of items. For example, a bed is an item for lying down in the adults' perspective, while for children it is an item for jumping up and down. Additionally, imagination is a way of problem creation. While machine

[1] https://openai.com/

learning has shown promising results in playing Atari games (Silver et al., 2016), it still can't create a new game (Mahadevan, 2018).

The notion of creativity in Artificial Intelligence is, however not new (Boden, 1998), but it has recently gained interest, perhaps because of the improved capabilities of deep networks. For example (Ha & Schmidhuber, 2018) used Generative Adversarial Networks (GAN) that learns to generate new unseen data from a given data. In particular, they trained a car in openAI car racing environment as well as training a network to generate similar scenarios constituting a kind of off-line "dream world". The trained car was then trained in a generated situation by the GAN model as a dreaming world. The works showed that the agent performed higher when trained in the generated unseen environment and then injected back to the real environment. However, this technique lacks the availability of physical dimension of the environment. Generating simulated world using similar images doesn't imply embedding the physics of this world in it. In order to have more effective learning by dreaming, our approach uses a dynamic physical simulation. This simulation should have both abilities of generating new situations as well as the physics simulation. Related work done by the OpenAI team has shown that a robot arm can learn in simulation and then highly perform in the real world (Duan et al., 2017). Duan et al. trained a robot arm in simulation using learning by demonstration to build a block tower. The robot arm picks blocks from different locations and then stacks them horizontally to form a tower. The trained agent was then tested with real blocks and proved to perform similarly to the simulation. Our approach to learning by dreaming is outlined in the following paragraphs. Sutton (1991) proposed the Dyna architecture which includes both reinforcement learning and learning from previous experience. In Dyna architecture, the agent memorizes the reinforcement learning components of state, action, reword and next state in memory then repeat this memory in supervised learning. This approach was introduced in Deep Q Learning as a replay mechanism (Mnih et al., 2013). The replay is a buffer of previous experience that the agent re-practice while it learns new experience in reinforcement learning. Although the Dyna architecture or the replay buffer mechanism can be considered as imagination learning, it restricts the imagination by the previous states and actions that the agent has tried. The imagination technique should also allow the agent to explore unexperienced states. In order to have more effective learning by dreaming, our approach uses a dynamic physical simulation. This simulation should have both abilities of generating new situations as well as the physics simulation. Related work done by the OpenAI team has shown that a robot arm can learn in simulation and then highly perform in the real world (Duan et al., 2017). Duan et al. trained a robot arm in simulation using learning by demonstration to build a block tower. The robot arm picks blocks from different locations and then stacks them horizontally to form a tower. The trained agent was then tested with real blocks and proved to perform similarly to the simulation. Our approach to learning by dreaming is outlined in the following paragraphs.

The system consists of the following parts (as also shown in *Figure 1*):

- **OpenDS:** a dynamic game tool engine that simulates driving in a road taking into account the physics relying on this. The unique function of OpenDS (opends.dfki.de) is that it is a dynamic simulation which means that it creates a simulated road from a road description without explicitly program the details of the road. The created road is the environment that the agent interacts with to know the current state and send the taken action. The environment sends the current state as a scenario message 20 messages per second. When the environment gets the maneuver message it calculates the new car position and renders the car to the new state. As such the car simulator is a kind of container for the episodic simulation.

- **Co-driver agent:** a learning agent that learns in an interactive way through deep reinforcement learning. The co-driving receives the current state as a reinforcement learning state. Then the agent chooses the best action according to a policy and sends the selected action. Due to the high dimensionality of having continuous actions, the range of actions were discretized into four actions: right, left, accelerate and break. According to the action taken, the co-driver receives a reward that represents how good the action is in the given state. The reward could either be a punishment for a wrong move or a positive reward for fulfilling the needs. For this experiment the main task is lane keeping. For the sake of simplicity, a tabular method is used for the reinforcement learning. In this method, the space of states is discretized into all the possible states with all the possible actions and assign expected reward for each.

- **Middleware connector:** a layer responsible for connecting the learning agent to the car simulator OpenDS. It receives the scenario message from OpenDS and converts it to reinforcement learning states and passes it to the co-driver agent. It then receives the selected action and creates a maneuver message which it passes it to the simulation. This layer is also responsible for determining the rewards that the agent takes for each step. The reward is calculated based on the lane keeping task. In addition, all the data is packed and passed to the episodic generator to keep track of the learning performance.

- **Episodic generator:** this layer is responsible for creating road descriptions as block components allowing openDS to render it into a physical simulation. The generator gets the performance data from the middleware connector, analyses it and find the week points where the agent struggled to learn. Accordingly, it generates new roads that focuses on the agent's weaknesses.

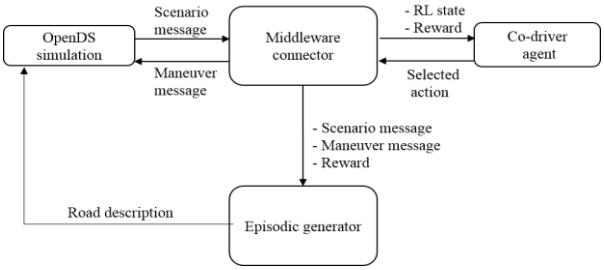

Figure 1 Episodic generator system architecture for self-driving car

From a process prospective the setup works as follows: First, the episodic generator creates road descriptions of 100 roads. These initial rewards are generated randomly from the available possible building blocks of the simulator to explore the possible roads. Next, OpenDS renders these descriptions into the physical road simulation and runs the first road. The middleware connector is responsible for connecting the simulation to the agent. At the beginning, the agent starts with no knowledge about driving, which means that all the initial space state pairs are initialized and the initial rewards are set to values higher than zero to stimulate the agent to explore new states. The training process then starts with the connection between the simulator OpenDS and the agent through the middleware connector. OpenDS sends the scenario message. The agent receives reinforcement learning states from the middleware connector after being manipulated from the scenario message. The agent matches this state with the states in the state-action pair table. The agent then chooses the action that is expected to receive the highest future reward as.

$$Q^*(s,a) = max_\pi Q^\pi(s,a)$$

When the agent receives the reward along with the new state, the agent updates the state-action pair table by:

$$Q^{new}(s_t,a_t) \leftarrow (1-\alpha).Q(s_t,a_t) + \alpha.(r_t + \gamma.maxQ(s_{t+1},a))$$

Where s_t is the current state, a_t is the action taken in the current state, α is the learning rate, r_t is the reward, γ is the discount factor.

When the agent drives out of the road, the agent gets high penalty (negative reward) and starts the road over. The simulation runs till the agent finishes all the roads successfully. During training, the log data including the road description, the driving performance, number of time the agent losses in a road, the accumulated reward and the distance driven are accumulated through roads. The episode generator analyses this data and uses it to generate new roads. The episode generator finds the roads that the agent struggled to pass (took more trials to finish) and generate similar features of this road. The new roads consist of the road blocks that the agent performed less well in. For example, if the agent poorly passed the sharp curves, the episode generator creates road descriptions that includes sharp roads.

Acknowledgement. This work has been supported by funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731593.

References

Boden, M. A. (1998). Creativity and artificial intelligence. *Artificial Intelligence*, *103*(1–2), 347–356.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. *ArXiv Preprint ArXiv:1606.01540*.

Duan, Y., Andrychowicz, M., Stadie, B., Ho, O. J., Schneider, J., Sutskever, I., … Zaremba, W. (2017). One-shot imitation learning. In *Advances in neural information processing systems* (pp. 1087–1098).

Ha, D., & Schmidhuber, J. (2018). World Models. *ArXiv Preprint ArXiv:1803.10122*.

Hesslow, G. (2002). Conscious thought as simulation of behaviour and perception. *Trends in Cognitive Sciences*, *6*(6), 242–247. https://doi.org/10.1016/S1364-6613(02)01913-7

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., … Wierstra, D. (2015). Continuous control with deep reinforcement learning. *ArXiv Preprint ArXiv:1509.02971*.

Lio, M. D., Mazzalai, A., Windridge, D., Thill, S., Svensson, H., Yüksel, M., … Heich, H. J. (2017). Exploiting dream-like simulation mechanisms to develop safer agents for automated driving: The "Dreams4Cars" EU research and innovation action. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1–6). https://doi.org/10.1109/ITSC.2017.8317649

Mahadevan, S. (2018). Imagination Machines: A New Challenge for Artificial Intelligence. *AAAI Conference*.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *ArXiv Preprint ArXiv:1312.5602*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., … Lanctot, M. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Singh, S., Okun, A., & Jackson, A. (2017). Artificial intelligence: Learning to play Go from scratch. *Nature*, *550*(7676), 336.

Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, *2*(4), 160–163.

Svensson, H. (2013). *Simulations*. Linköping: Linköping University Electronic Press. Retrieved from http://www.diva-portal.org/smash/record.jsf?pid=diva2:658266

Svensson, H., Thill, S., & Ziemke, T. (2013). Dreaming of electric sheep? Exploring the functions of dream-like mechanisms in the development of mental imagery simulations. *Adaptive Behavior*, *21*(4), 222–238.