Accelerating Reinforcement Learning in Engineering Systems

Tham Chen Khong
with contributions from Zhou Chongyu and Le Van Duc
Department of Electrical & Computer Engineering
National University of Singapore

Reinforcement Learning

• An agent learns to perform actions that maximize long term rewards
• Discovers solutions through interaction with the environment

• Different from supervised learning which a lot of deep learning is used for
• Deep learning can still be applied for RL, but target value changes
• RL is a powerful technique, but is usually computationally expensive and may be impractical since it requires many interactions with the environment
Mastering the game of Go without human knowledge

David Silver1*, Julian Schrittwieser1, Karen Simonyan1*, Joannis Antonoglou1, Aja Huang1, Arthur Guez1, Thomas Hubert1, Lucas Baker1, Matthew Lai1, Adrian Bolton1, Yutian Chen1, Timothy Lillicrap1, Fan Hui1, Laurent Sifre1, George van den Driessche1, Thore Graepel1 & Demis Hassabis1

A long-standing goal of artificial intelligence is an algorithm that learns, rubs shoulders, superhuman proficiency in challenging domains. Recently, AlphaGo became the first program to defeat a world champion in the game of Go. The tree search in AlphaGo evaluated positions and selected moves using deep neural networks. These neural networks were trained by supervised learning from human expert moves, and by reinforcement learning from self-play. Here we introduce an algorithm based solely on reinforcement learning without human data, guidance or domain knowledge beyond game rules. AlphaGo becomes its own teacher: a neural network is trained to predict AlphaGo’s own move selections and also the winner of AlphaGo’s games. This neural network improves the strength of the tree search, resulting in higher quality move selection and stronger self-play in the next iteration. Starring rubs shoulders, our new program AlphaGoZero achieved superhuman performance, winning 100–0 against the previously published, champion-defeating AlphaGo.

Human-level Control in Atari 2600 Games

Mnih et al, 2013/15

Deep Q-Network (DQN)
Engineering Systems

- Complex systems
  - behaviour described by differential equations
  - may have a number of operational modes
- Real-time control is required
- Need to operate reliably and safely
- Examples
  - vehicles, robots, manufacturing and communication and computing systems

Execution Time:
CPU vs GPU on OpenAI Gym Tasks

**CPU**: Intel Core i7-3770 CPU 3.40 GHz (4 cores, 8 GB RAM) (power consumption: 47-100 W)

**GPU**: NVIDIA GeForce GTX 1060 (1,280 cores, 6 GB) (≈$350; power consumption: 120 W)

*Training parameters:*
- # Episodes: 1,000
- # Iterations in each Episode: 500
Computation Offloading in an Ad-hoc Mobile Cloud (5G - IoT - Fog Computing operation scenario)

- Consider an ad-hoc mobile cloud where a mobile user can offload its computation tasks to nearby mobile cloudlets (e.g., smart phones and vehicles) with available computation resources
- Computation tasks are offloaded to the cloudlets via a device-to-device (D2D) communication-enabled cellular network
- Deep reinforcement learning (DRL) is applied for the user to learn an optimal offloading policy with the objective of maximizing the user’s utility, while minimizing the required payment, energy consumption, processing delay and task loss

D V Le, and C K Tham, A Deep Reinforcement Learning (DRL)-based Offloading Scheme in Ad-hoc Mobile Clouds, submitted to *IEEE PerCom 2018*
DQN-based Offloading Decision Learning

- The user maintains a neural network to estimate Q-values for all pairs of states and actions
- In the deep Q-network, the experience replay and target network techniques are used to train the neural network at every learning step
- The $\epsilon$-greedy policy is adopted to select an action for each state based on the estimated Q-values

Offloading Results

Achieves good payment, energy consumption, processing delay and task loss performance

Training parameters:
# Episodes: 100
# Iterations in each Episode: 5,000
$\epsilon$ is decreased from 0.99 to 0.1 with decay rate 0.9999

CPU: Intel Core i7-3770 CPU 3.40 GHz (4 cores, 16 GB RAM)
GPU: NVIDIA GeForce GTX 1060 (1,280 cores, 6 GB)
Robot Manipulator Control

• Goal: move end-effector from any starting position to the destination position without colliding with any obstacle
• Generate torque commands to drive each link
• A dynamical system whose model is governed by differential equations

Train DQN

1. Initialize
2. Start from random location
3. Simulate (arms move)
4. Reach/collision/100 steps
5. Repeat 5000 trials
6. Repeat 200 cycles
7. Update model (replay)
Robot Manipulator Results

Is this task too difficult for Reinforcement Learning?
No. It can be solved ... using a different function approximator

Faster
Higher resolution

Composite Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>Decomposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>reach 1</td>
<td>$T_1$</td>
</tr>
<tr>
<td>$T_2$</td>
<td>reach 2</td>
<td>$T_2$</td>
</tr>
<tr>
<td>$T_3$</td>
<td>reach 3</td>
<td>$T_3$</td>
</tr>
<tr>
<td>$C_1$</td>
<td>reach 1, 3</td>
<td>$T_1 T_3$</td>
</tr>
<tr>
<td>$C_2$</td>
<td>reach 2, 3</td>
<td>$T_2 T_3$</td>
</tr>
<tr>
<td>$C_3$</td>
<td>reach 1, 2, 3</td>
<td>$T_1 T_2 T_3$</td>
</tr>
</tbody>
</table>
Conclusion

• Reinforcement Learning is capable of solving engineering tasks
• Hardware accelerators like GPUs* and algorithmic improvements have enabled RL to be accelerated in time and reduce the number of iterations or interactions with the environment that are required
• However, direct application of deep learning techniques may not yield the best results
• Further research into various aspects is needed to enhance RL’s ability to better solve engineering problems

Thank You
Questions?
Contact: Tham Chen Khong
E-mail: eletck@nus.edu.sg