Deep Reinforcement Learning
Building Blocks

Arjun Chandra
Research Scientist
Telenor Research / Telenor-NTNU AI Lab
arjun.chandra@telenor.com
@boelger

8 November 2017

https://join.slack.com/t/deep-rl-tutorial/signup
The Plan

• The Problem
• (deep) RL Concepts by Example
• Problem Decomposition
• Solution Methods
  • Value Based
  • Policy Based
  • Actor-Critic
how to make decisions over time to maximise my return / “long term reward”?
emergence of locomotion

https://deepmind.com/blog/producing-flexible-behaviours-simulated-environments/
https://www.youtube.com/watch?v=hx_bgoTF7bs
As we know...

late 1980s

RL for robots using NNs, L-J Lin. PhD 1993, CMU

Gerald Tesauro

1995

2004

Stanford

http://heli.stanford.edu/

2013 —

Vlad Mnih et. al.

2015 —

Google DeepMind

David Silver et. al.

http://heli.stanford.edu/
Problem Characteristics

- **dynamic**
- **uncertainty/volatility**
- uncharted/unimagined/
  exception laden
- **delayed** consequences
- requires **strategy**

Solution

machine with agency which learn, plan, and act to find a strategy for solving the problem autonomously to some extent probe and learn from feedback focus on the long-term objective explore and exploit
what is the sequence of actions I could take to maximise my return / “long term reward”?
Reinforcement Learning

- Observation and feedback on actions
- Problem/Environment
- Agent
  - Model
  - Goal
  - \( \pi/Q \)
- Action
- Model dynamics model
- \( \pi/Q \) policy/value function
- Goal: maximise return \( E(R) \)
the excruciatingly awesome MDP game!

Inspired by Rich Sutton's tutorial:
https://www.youtube.com/watch?v=ggqnxyjaKe4
the MDP \((S, A, P, R, \gamma)\)

- **R**: immediate reward function \(R(s, a)\)
- **P**: state transition probability \(P(s'|s, a)\)

---

Image of a Markov Decision Process (MDP) with states A and B, and transitions labeled with rewards and probabilities.
the problem
(cartoon of an MPD)

reward

state

action
agent’s job/goal?

maximise expected cumulative reward/return
toy problem

home

Diagram showing a grid with a dog and a cat, indicating movement between different states.
state and action spaces

- size of these spaces can be quite large
- specifying the spaces is crucial in designing a good learning agent
5 integer values between 1 and 100: \{22, 44, 12, 67, 9\}

size of state space = 100 \times 100 \times 100 \times 100 \times 100 \times 100

can quantise state space differently

5 values belonging to 2 classes: \{1, 2, 1, 2, 1\}

size of state space = 2 \times 2 \times 2 \times 2 \times 2

in the toy problem? 9
taking an action in some state results in an immediate reward (can be negative)
home
<table>
<thead>
<tr>
<th>home</th>
<th>what if the cat were to start here some other day?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

- Image of a cat at the top left corner of the table.
reward system should tell the agent:
what to achieve
rather than how to achieve
reward

this is all the feedback an agent gets!

immediate!
but agent has to choose an action based on expected return
expected return?
task

- episodic (there is an end)
- continual (there is no end)
episodic

(there is an **end**)  

agent taking **finite (say 5) steps** till the end...

should act based on the  

e.g. average of the following 

\[ R_0 = r_1 + r_2 + r_3 + r_4 + r_5 \]
continual
(there is no end)

agent can continue acting for infinite steps in time...

should discount future rewards and act based on e.g. average of the following

\[ R_0 = r_1 + \gamma r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \gamma^4 r_5 + \cdots \]
discount

future reward is probably more uncertain than immediate reward

shortsighted?  \( \gamma = 0 \)  

farsighted?  \( \gamma = 1 \)  

\[
R_0 = r_1 + \gamma r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \gamma^4 r_5 + \cdots
\]
\begin{align*}
R_0 &= \sum_{k=0}^{T} \gamma^k r_{k+1} \\
E\{R_t\} &= \sum_{k=0}^{T} \gamma^k r_{t+k+1}
\end{align*}
immediate reward

further reward possibilities
\[ E \{ R_t \} = \sum_{k=0}^{T} \gamma^k r_{t+k+1} \]
but these expected returns are not known to agent beforehand!
what knowledge might the agent try to acquire to behave properly?
rank/probability of an action in some state bringing max expected return (long term value)?
expected long term value of being in each state, under some action selection scheme?

<table>
<thead>
<tr>
<th></th>
<th>E{R}</th>
<th>E{R}</th>
<th>E{R}</th>
</tr>
</thead>
<tbody>
<tr>
<td>E{R}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E{R}</td>
<td></td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>home</td>
<td></td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
</tbody>
</table>
expected long term value of taking some action in each state, then behaving using some action selection scheme?

<table>
<thead>
<tr>
<th></th>
<th>E{R}</th>
<th>E{R}</th>
<th>E{R}</th>
</tr>
</thead>
<tbody>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
<tr>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
<td>E{R}</td>
</tr>
</tbody>
</table>
modelling dynamics / mapping the environment?

If I go South, I will meet
prediction problem
learn to predict expected long term reward/value

control problem
learn to find the optimal action selection scheme/policy
policy: action selection
value: how good is an action/state
model: predict next state/reward to look ahead/plan
types of RL agents?

- **Value based**
  - Both value and policy

- **Policy based**
  - Value/policy
  - Model of dynamics
we will focus on value based RL in the first half
action selection?

Expected return for carrying out an action is its value.

Values of each possible action in the current state helps select actions!
policy can be derived from value (e.g. act greedily)
but what are these values?

<<expected returns unknown>>

<<actions based on unknowns>>
value can be **estimated** by sampling environment while acting using some policy

e.g. act, accumulate new reward (ground truth), and update
agent maintains values for actions within each state

selects actions using these values under some “policy”
agent **maintains state values**

selects actions using these values under some **“policy”**

but... agent needs a **model of the environment!**
home

9 states

10^{16992} (pixels)
10^{308} (ram)

continuous!
extract features that help generalise across states
Action values given state

Q(s, north)  Q(s, south)  Q(s, east)  Q(s, west)

State s

Features
"Q V\{E|R_{t}\}"
policy?

probability of choosing an action in state/feature representation thereof

\[ \pi \quad Q^\pi(s,a) \quad V^\pi(s) \]
usual policies

- greedy: choose best action
- \(\varepsilon\)-greedy: choose best action with probability \(1-\varepsilon\)
- soft-max: choose action with probability given by its value
exploration vs. exploitation
trial and error

game play: try new moves
ads: try new ads
a/b testing:
try new website feature
smart camera networks:
try new comm. protocol

Static, dynamic and adaptive heterogeneity in socio-economic
distributed smart camera networks, P. R. Lewis, L. Esterle, A. Chandra,
B. Rinner, J. Torresen, and X. Yao, ACM Transactions on Autonomous
$V^*(s) = \max_\pi V^\pi(s)$

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

$\pi^*(a|s) = \begin{cases} 
1 & \text{if } a = \arg\max_a Q^*(s,a) \\
0 & \text{otherwise}
\end{cases}$
estimation?

<<use currently visible returns to update values of where you are coming from>>

the current state (or state-action pair) has an estimated value (say zero/random initially),

which can be used together with $r_{t+1}$ to update value of previous state (or state-action pair)
\[ \text{i.e.} \]

\[
\frac{\text{fraction of } (\text{currently visible returns} - \text{old value})}{\text{old value}} + (1-\text{fraction}) \text{ old value} + \text{fraction} \text{ curr. vis. returns} = \text{new value}
\]
immediate reward $r_{t+1}$

further reward possibilities

$R_t = \sum_{k=0}^{T} \gamma^k r_{t+k+1}$

$E\{R_t\}$

$r_{t+1} + \gamma Q(s',a')$

$r_{t+1} + \gamma Q(s_{t+1},a_{t+1})$

$r_{t+1} + \gamma E\{R_{t+1}\}$
\[ V(s) \leftarrow V(s) + \alpha \left( r_s^a + \gamma V(s') - V(s) \right) \]

\[ Q(s, a) \leftarrow Q(s, a) + \alpha \left( r_s^a + \gamma Q(s', a') - Q(s, a) \right) \]
e.g. update
a lookup table maintaining expected returns

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>7</td>
<td>8</td>
<td>7</td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n</td>
<td></td>
<td></td>
<td>7</td>
</tr>
</tbody>
</table>
$Q(s,a) \leftarrow Q(s,a) + \alpha (r_s^a + \gamma Q(s',a') - Q(s,a))$

let's play with a version of the above update rule:

$Q(s,a) \leftarrow Q(s,a) + \alpha (r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a))$
indicates $a'$ to be the action with maximum value in next state $s'$

let's play with a version of the above update rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a))$$
our toy problem

lookup table

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>S</th>
<th>E</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

home
our toy problem
lookup table
reward structure?

move...
to any cell except 5 and 7: -1
out of bounds: -5
to 5: -10
to 7/home: 10
let’s fix $\alpha = 0.1$, $\gamma = 0.5$
\[ Q(s,a) \leftarrow Q(s,a) + \alpha(r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a)) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]

say \( \varepsilon \)-greedy policy...

episode 1 begins...
$Q(s, a) \leftarrow Q(s, a) + \alpha [r_s^a + \gamma \max_a Q(s', a') - Q(s, a)]$

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>?</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td></td>
<td>5</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td></td>
<td>8</td>
<td>0</td>
<td>9</td>
</tr>
</tbody>
</table>

α = 0.1  
γ = 0.5  

-1
\[ Q(s,a) \leftarrow Q(s,a) + \alpha \left( r_s^a + \gamma \max_a Q(s',a') - Q(s,a) \right) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>home</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
\[ Q(s,a) \leftarrow Q(s,a) + \alpha \left( r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a) \right) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
\[ Q(s,a) \leftarrow Q(s,a) + \alpha \left( r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a) \right) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
\[
Q(s,a) \leftarrow Q(s,a) + \alpha \left( r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a) \right)
\]

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>-0.1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>home</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\(\alpha = 0.1\)
\(\gamma = 0.5\)
$Q(s,a) \leftarrow Q(s,a) + \alpha (r_s^a + \gamma \max_a Q(s',a') - Q(s,a))$

\[
\begin{array}{ccc}
-0.5 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 7 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

\[\alpha = 0.1 \quad \gamma = 0.5\]
\[ Q(s,a) \leftarrow Q(s,a) + \alpha(r_s^a + \gamma \max_a Q(s',a') - Q(s,a)) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
\[ Q(s,a) \leftarrow Q(s,a) + \alpha (r^a_s + \gamma \max_a Q(s',a') - Q(s,a)) \]

\[ \alpha = 0.1 \]

\[ \gamma = 0.5 \]
\( Q(s,a) \leftarrow Q(s,a) + \alpha \left[ r_s^a + \gamma \max_a Q(s',a') - Q(s,a) \right] \)

\[
\begin{array}{ccc}
-0.5 & 0 & 0 \\
0 & 1 & 0 \\
-0.1 & 2 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

\( \alpha = 0.1 \)
\( \gamma = 0.5 \)

*home*
\[
Q(s,a) \leftarrow Q(s,a) + \alpha(r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a))
\]

\begin{array}{ccc}
  & 0 & \text{home} \\
-0.5 & 0 & 1 \\
0 & -0.1 & 2 \\
-0.1 & 0 & 3 \\
0 & 0 & 4 \\
0 & 0 & 5 \\
0 & 0 & 0 \\
\end{array}

\begin{array}{ccc}
  & 0 & \text{home} \\
-0.5 & 0 & 1 \\
0 & -0.1 & 2 \\
-0.1 & 0 & 3 \\
0 & 0 & 4 \\
0 & 0 & 5 \\
0 & 0 & 0 \\
\end{array}

\alpha = 0.1 \\
\gamma = 0.5
\[ Q(s, a) \leftarrow Q(s, a) + \alpha [r^a_s + \gamma \max_{a'} Q(s', a') - Q(s, a)] \]

\[ \alpha = 0.1 \quad \gamma = 0.5 \]
\[ Q(s,a) \leftarrow Q(s,a) + \alpha (r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a)) \]

\[ \alpha = 0.1 \quad \gamma = 0.5 \]
$$Q(s,a) \leftarrow Q(s,a) + \alpha (r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

<table>
<thead>
<tr>
<th></th>
<th>-0.5</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>-0.1</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>home</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$$\alpha = 0.1$$

$$\gamma = 0.5$$
\[ Q(s,a) \leftarrow Q(s,a) + \alpha (r^a_s + \gamma \max_a Q(s',a') - Q(s,a)) \]

\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
$$Q(s,a) ← Q(s,a) + \alpha (r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\(\alpha = 0.1\)
\(\gamma = 0.5\)
\[ Q(s,a) \leftarrow Q(s,a) + \alpha(r^a_s + \gamma \max_{a'} Q(s',a') - Q(s,a)) \]

<table>
<thead>
<tr>
<th></th>
<th>-0.5</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>-0.1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>-1</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

\( \alpha = 0.1 \)
\( \gamma = 0.5 \)
\[ Q(s,a) \leftarrow Q(s,a) + \alpha(r_s^a + \gamma \max_{a'} Q(s',a') - Q(s,a)) \]

<table>
<thead>
<tr>
<th></th>
<th>-0.5</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
</tbody>
</table>

\( \alpha = 0.1 \)
\( \gamma = 0.5 \)

episode 1 ends.
let’s work out the next episode, starting at state 4

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>-0.1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

home

$\alpha = 0.1$
$\gamma = 0.5$

go WEST and then SOUTH

how does the table change?
\[ \alpha = 0.1 \]
\[ \gamma = 0.5 \]
and the next episode, starting at state 3

go WEST -> SOUTH -> WEST -> SOUTH
over time, values will converge to optimal!
what we just saw was some episodes of Q-learning

values update towards value of optimal policy:

target comes from value of assumed next best action

off-policy learning
SARSA-learning?

Values update towards value of *current policy*: target comes from value of *the actual next action*.

On-policy learning
data not generated by target policy

\( \epsilon: 0.1 \)
\( \gamma: 1.0 \)

data generated by target policy

Example credit Travis DeWolf: https://studywolf.wordpress.com/ and https://git.io/vFBvv
Problem Decomposition

nested sub-problems

solution to sub-problem informs solution to whole problem
Bellman Expectation Backup

system of linear equations
solution: value of policy

Bellman expectation equations under a given policy

\[ v_\pi(s) = \sum_a \pi(a|s) \left( r_s^a + \gamma \sum_{s'} P_{ss'}^a v_\pi(s') \right) \]

\[ q_\pi(s,a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a'|s') q_\pi(s',a') \]
Bellman Optimality Equations

Value of $s, v(s)$ = $P(path) \times Value(path)$

Value of $s', v(s')$ = $P(path) \times Value(path)$

Bellman optimality equations under optimal policy

$v_*(s) = \max_a \left( r_s^a + \gamma \sum_{s'} P_{ss'}^a v_*(s') \right)$

$q_*(s,a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \max_{a'} q_*(s',a')$
Value Based
Dynamic Programming

...using Bellman equations as iterative updates

what’s best to do?
Dynamic Programming
...using Bellman equations as iterative updates

what’s best to do?

home
Policy Iteration

Evaluate Policy
(sweep, apply Bellman expectation)

Improve Policy
(greedy)

Value of $\pi = P(\text{path}) \times \text{Value(\text{path})}$

$$q_{\pi}(s, a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a' | s') q_{\pi}(s', a')$$
\[ q_\pi(s, a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a'| s') q_\pi(s', a') \]

\[ \pi(S|1): 1.0 \text{ (greedy)} \]
\[ \pi(W|2): 1.0 \text{ (greedy)} \]

iteratively apply Bellman expectation equations in inner loop until values do not change much

use greedy policy, given new values
Value Iteration

Find Optimal Value and Policy
(sweep, apply Bellman optimality)

\[ q^*(s,a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \max_{a'} q^*(s',a') \]
Value of \( path \) = \( P(path) \times \text{Value}(path) \)

\[ q_*(s,a) = r_s^a + \gamma \sum_{s'} P_{ss'}^a \max_a q_*(s',a') \]

iteratively apply Bellman optimality equations until values do not change much
Bellman backups

largest distance between values decreases after Bellman backups
From DP to Learning

full-width backups to sample backups
Backup with Sample Return
Backup with Guess
Incremental Updates

\[ E \{ R \} \approx \mu_k = \frac{1}{k} \sum_{\tau=1}^{k} R_{\tau} \]

\[ \mu_k = \mu_{k-1} + \frac{1}{k} \left( R_k - \mu_{k-1} \right) \]  

batched

\[ \mu_k = \mu_{k-1} + \alpha \left( R_k - \mu_{k-1} \right) \]  

incremental

running (saw this in Q-learning!)
Sample and Bootstrap

- Bootstrapping, $\lambda$
- Shallow backup
- Deep backup
- Sample backup
- Full-width backup
- Dynamic prg.
- Step returns/guess
- Exhaustive search
- Full trajectory returns
- Deep backup
- Bootstrapping, $\lambda$
It all comes down to:

estimating returns

optimising

towards achieving returns
Q-learning

full-width backups to sample backups

target policy optimal
SARSA

full-width backups
to sample backups

target policy
same as
behaviour policy
scaling up RL with function approximation
Approximate Q-learning

e.g. linear approximation

\[ Q_\theta(s,a) = \theta_0 f_0(s,a) + \theta_1 f_1(s,a) + \ldots + \theta_n f_n(s,a) \]

\[ Q_{\text{target}} = (r^a_s + \gamma \max_{a'} Q(s',a')) \]

\[ \theta \leftarrow \theta - \alpha \nabla_\theta \frac{1}{2} \left( Q_{\text{target}} - Q_\theta(s,a) \right)^2 \]
gradient updates equivalent to tabular Q updates

Say $\theta \in \mathbb{R}^{\mathcal{|S|\times|A|}}$, so $Q_\theta(s,a) = \theta_{sa}$

$Q_{\text{target}} = r_s^a + \gamma \max_{a'} Q(s',a')$

$\theta_{sa} \leftarrow \theta_{sa} - \alpha \nabla \frac{1}{2} \left( Q_{\text{target}} - \theta_{sa} \right)^2$

$\theta_{sa} \leftarrow \theta_{sa} - \alpha \left( -Q_{\text{target}} + \theta_{sa} \right)$

$\theta_{sa} \leftarrow \theta_{sa} + \alpha \left( Q_{\text{target}} - \theta_{sa} \right)$

$\theta_{sa} \leftarrow (1-\alpha) \theta_{sa} + \alpha Q_{\text{target}}$
Human-level control through deep reinforcement learning, Mnih et. al., Nature 518, Feb 2015
human level game control

- **pixel** input
- **18 joystick/button positions** output
- **change in game score** as feedback
- **convolutional net representing Q**
- **backpropagation** for training!

Human-level control through deep reinforcement learning,
Mnih et. al., Nature 518, Feb 2015
http://www.nature.com/nature/journal/v518/n7540/full/nature14236.html
neural network
backpropagation

What is the **target** against which to minimise error?

\[
\mathcal{L}(w) = \mathbb{E}\left[ \left( r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w) \right)^2 \right]
\]

\[
\frac{\partial \mathcal{L}(w)}{\partial w} = \mathbb{E}\left[ \left( r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w) \right) \frac{\partial Q(s, a, w)}{\partial w} \right]
\]
experience replay buffer

save transition in memory

randomly sample from memory for training = i.i.d
\[ (r + \gamma \max_{a'} Q(s', a', w^-) - Q(s, a, w))^2 \]

Human-level control through deep reinforcement learning, Mnih et. al., Nature 518, Feb 2015
however
training is
SLOOo0o0o0o.... W
parallelise...
Parallel Asynchronous Training

value and policy based methods

parallel agents  shared parameters  lock-free updates

Asynchronous Methods for Deep Reinforcement Learning, Mnih et. al., ICML 2016
parallel learners

shared params

HOGWILD! updates

Agent

params

HOGWILD! updates

https://github.com/traai/async-deep-rl
Policy Based
$\pi$ state $s$

features

policy $\pi(a|s)$

$\pi(\text{north}|s)$
$\pi(\text{south}|s)$
$\pi(\text{east}|s)$
$\pi(\text{west}|s)$
Intuition

\[ \tau : s_1, a_1, r_1^1, s_2, a_2, r_2^2, \ldots, s_{H-1}, a_{H-1}, r_{H-1}^{H-1} \]

\[ R\tau_1 = 10 \]

\[ R\tau_2 = 5 \]

\[ R\tau_3 = 2 \]
Intuition

$\pi(a|s)$ along path with high return higher

$\tau: s_1, a_1, r_1^1, s_2, a_2, r_2^2, \ldots, s_{H-1}, a_{H-1}, r_{H-1}^{H-1}$

Probabilities are relative

$R\tau_1 = 10$

$R\tau_2 = 5$

$R\tau_3 = 2$
Revisiting the Objective

\[
\tau : s_1, a_1, r_1^1, s_2, a_2, r_2^2, \ldots, s_{H-1}, a_{H-1}, r_{H-1}^{H-1}
\]

\[
\max_\theta E_{\tau} \left\{ \sum_{t=0}^{H-1} r_{s_t}^a \mid \pi_\theta \right\}
\]

\[
\max_\theta J(\theta) = \max_\theta \sum_\tau P(\tau \mid \theta) R(\tau)
\]
Samples → Gradient

\[ J(\theta) = \sum_{\tau} P(\tau | \theta) R(\tau) \]

\[ \max_{\theta} J(\theta) \]

\[ \theta \leftarrow \theta + \nabla_{\theta} J(\theta) \]

\[ \nabla_{\theta} J(\theta) = \nabla_{\theta} \sum_{\tau} P(\tau | \theta) R(\tau) \]

\[ = \sum_{\tau} \nabla_{\theta} P(\tau | \theta) R(\tau) \]

\[ = \sum_{\tau} \frac{P(\tau | \theta)}{P(\tau | \theta)} \nabla_{\theta} P(\tau | \theta) R(\tau) \]

\[ = \sum_{\tau} P(\tau | \theta) \frac{\nabla_{\theta} P(\tau | \theta)}{P(\tau | \theta)} R(\tau) \]

\[ = \sum_{\tau} P(\tau | \theta) \nabla_{\theta} \log P(\tau | \theta) R(\tau) \]

\[ \nabla_{\theta} J(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \nabla_{\theta} \log P(\tau^{(i)} | \theta) R(\tau^{(i)}) \]
\[ \nabla_{\theta} \log P(\tau | \theta) = \nabla_{\theta} \log \left[ \prod_{t=0}^{H-1} P(s_{t+1} | s_t, a_t) \cdot \pi_\theta(a_t | s_t) \right] \]

\[ = \nabla_{\theta} \left[ \sum_{t=0}^{H-1} \log P(s_{t+1} | s_t, a_t) + \sum_{t=0}^{H-1} \log \pi_\theta(a_t | s_t) \right] \]

\[ = \nabla_{\theta} \sum_{t=0}^{H-1} \log \pi_\theta(a_t | s_t) = \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_\theta(a_t | s_t) \]

\[ \nabla_{\theta} J(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \left( \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_\theta(a_t^{(i)} | s_t^{(i)}) \right) R(\tau^{(i)}) \]
\[ \nabla_\theta J(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \left( \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)}) \right) R(\tau^{(i)}) \]

For each action \( a_t \) in state \( s_t \) during each trajectory \( m \)

\[ \nabla_\theta \log \pi_\theta(a_t \mid s_t) R(\tau) \]

\( \Delta \theta \) to increase \( \pi_\theta(a_t \mid s_t) \times \)
Noisy Gradient

$\Delta \theta$ to increase $\pi_\theta(a_t \mid s_t) \times$
Reduce Noise

$\Delta \theta$ to increase $\pi_\theta(a_t \mid s_t)$ x

$R(\tau_t \text{ onwards})$

$R$ time $t$ onwards

$R$ time $t$ onwards
Reduce Noise

\[ \Delta \theta \text{ to increase } \pi_\theta(a_t \mid s_t) \times \]

baseline \( b \)

(how much is action better than average)

\[ V = E\{R \mid s\} \]
Reduce Noise

$\Delta \theta$ to increase $\pi_\theta (a_t \mid s_t) \times \text{baseline } b$

(how much is action better than average)

$V = E\{R \mid s\}$
Actor-Critic
Reduce Noise

\[ \Delta \theta \text{ to increase } \pi_{\theta}(a_t \mid s_t) \]

**critic** \( Q \)

(expected long term value of action)

\[ Q = E\{R \mid s, a\} = E\{r + \gamma V\} \]
Reduce Noise

$$\Delta \theta$$ to increase $$\pi_\theta(a_t \mid s_t) \times A(s_t, a_t)$$

$$A = Q - V$$

(advantage of an action)

(how much is action better than average)
parallelise...
Parallel Asynchronous Training

value and policy based methods

parallel agents
shared parameters
lock-free updates

Asynchronous Methods for Deep Reinforcement Learning, Mnih et. al., ICML 2016
parallel learners

HOGWILD!

updates

shared params

Agent

https://github.com/traai/async-deep-rl
PAAC
(Parallel Advantage Actor-Critic)

Efficient Parallel Methods for Deep Reinforcement Learning,
A. V. Clemente, H. N. Castejón, and A. Chandra, RLDM 2017

https://github.com/alfredvc/paac

1 GPU/CPU
Reduced training time
SOTA performance

Alfredo Clemente
code for you to play with...


Telenor’s implementation of asynchronous parallel methods: https://github.com/traai/async-deep-rl

Alfredo’s faster parallel methods: https://github.com/alfredvc/paac

++...
Inspired to code/apply RL?

Next lecture:
Applications (and some hacking)
November 21, 2017

https://join.slack.com/t/deep-rl-tutorial/signup