Model Based Reinforcement Learning

Presenter: Adrian Hoffmann
Basic Set-Up
Basic Set-Up

\[ \text{state } S_t \quad \text{reward } R_t \quad \text{action } A_t \]

\[
\text{Environment} \quad \text{Agent}
\]

\[ R_{t+1} \quad S_{t+1} \]
Overview

• Definitions and their problems
• Why I want a model
• Paper “The Effect of Planning Shape on Dyna-style Planning in High-dimensional State Spaces”
• Paper “World Models”
Definition Model-Based
Definition Model-Based

Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure \( d \)
2: Input: model \( m \)
3: Input: policy \( \pi \)
4: Input: predictions \( v \)
5: Input: environment \( \mathcal{E} \)
6: Get initial state \( s \leftarrow \mathcal{E} \)
7: for iteration \( \in \{1, 2, \ldots, K\} \) do
8:     for interaction \( \in \{1, 2, \ldots, M\} \) do
9:         for planning step \( \in \{1, 2, \ldots, P\} \) do
10:            
11:        end for
12:    end for
13: end for

Interactions with the real Environment

Take advantage of the model

Van Hasselt et al. 2019
Definition Model-Based

Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure \( d \)
2: Input: model \( m \)
3: Input: policy \( \pi \)
4: Input: predictions \( v \)
5: Input: environment \( \mathcal{E} \)
6: Get initial state \( s \leftarrow \mathcal{E} \)
7: for iteration \( \in \{1, 2, \ldots, K\} \) do
8: \hspace{1em} for interaction \( \in \{1, 2, \ldots, M\} \) do
9: \hspace{2em} for planning step \( \in \{1, 2, \ldots, P\} \) do
10: \hspace{3em} end for
11: \hspace{1em} end for
12: \hspace{1em} end for
13: end for
Interlude – Model

- Statistical Models
- Gaussian Process
- RNN, LSTM
- (Variational Auto) Encoders
Definition Model-Based

Van Hasselt et al. 2019

Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure \(d\)
2: Input: model \(m\)
3: Input: policy \(\pi\)
4: Input: predictions \(v\)
5: Input: environment \(E\)
6: Get initial state \(s \leftarrow E\)
7: for iteration \(\in \{1, 2, \ldots, K\}\) do
8: for interaction \(\in \{1, 2, \ldots, M\}\) do
9: end for
10: for planning step \(\in \{1, 2, \ldots, P\}\) do
11: end for

Interactions with the real Environment

Take advantage of the model
## Definition Model-Based

**Algorithm 1** Model-based reinforcement learning

1. Input: state sample procedure \( d \)
2. Input: model \( m \)
3. Input: policy \( \pi \)
4. Input: predictions \( v \)
5. Input: environment \( \mathcal{E} \)
6. Get initial state \( s \leftarrow \mathcal{E} \)
7. for iteration \( \in \{1, 2, \ldots, K\} \) do
8.  
9.  for interaction \( \in \{1, 2, \ldots, M\} \) do
10.    Generate action: \( a \leftarrow \pi(s) \)
11.    Generate reward, next state: \( r, s' \leftarrow \mathcal{E}(a) \)
12.    \( m, d \leftarrow \text{UPDATEMODEL}(s, a, r, s') \)
13.    \( \pi, v \leftarrow \text{UPDATEAGENT}(s, a, r, s') \)
14.  end for
15. for planning step \( \in \{1, 2, \ldots, P\} \) do
16.  end for
17.
18.
19. end for
20.

Take advantage of the model

Van Hasselt et al. 2019
Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure $d$
2: Input: model $m$
3: Input: policy $\pi$
4: Input: predictions $v$
5: Input: environment $\mathcal{E}$
6: Get initial state $s \leftarrow \mathcal{E}$
7: for iteration $\in \{1, 2, \ldots, K\}$ do
8:     for interaction $\in \{1, 2, \ldots, M\}$ do
9:         Generate action: $a \leftarrow \pi(s)$
10:        Generate reward, next state: $r, s' \leftarrow \mathcal{E}(a)$
11:        $m, d \leftarrow \text{UPDATEMODEL}(s, a, r, s')$
12:        $\pi, v \leftarrow \text{UPDATEAGENT}(s, a, r, s')$
13:    end for
14:    Update current state: $s \leftarrow s'$
15:    end for
16:    for planning step $\in \{1, 2, \ldots, P\}$ do
17:        Generate state, action $\bar{s}, \bar{a} \leftarrow \bar{d}$
18:        Generate reward, next state: $\bar{r}, \bar{s}' \leftarrow m(\bar{s}, \bar{a})$
19:        $\bar{\pi}, \bar{v} \leftarrow \text{UPDATEAGENT}(\bar{s}, \bar{a}, \bar{r}, \bar{s}')$
20:    end for

Van Hasselt et al. 2019
Definition (purely) Model-Based

Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure $d$
2: Input: model $m$
3: Input: policy $\pi$
4: Input: predictions $v$
5: Input: environment $\mathcal{E}$
6: Get initial state $s \leftarrow \mathcal{E}$
7: for iteration $\in \{1, 2, \ldots, K\}$ do
8: for interaction $\in \{1, 2, \ldots, M\}$ do
9: Generate action: $a \leftarrow \pi(s)$
10: Generate reward, next state: $r, s' \leftarrow \mathcal{E}(a)$
11: $m, d \leftarrow \text{UPDATE MODEL}(s, a, r, s')$
12: Update current state: $s \leftarrow s'$
13: end for
14: for planning step $\in \{1, 2, \ldots, P\}$ do
15: Generate state, action $\bar{s}, \bar{a} \leftarrow d$
16: Generate reward, next state: $\bar{r}, \bar{s}' \leftarrow m(\bar{s}, \bar{a})$
17: $\pi, v \leftarrow \text{UPDATE AGENT}(\bar{s}, \bar{a}, \bar{r}, \bar{s}')$
18: end for
19: end for

Van Hasselt et al. 2019
Definition Model-Free
Definition Model-Based

Algorithm 1 Model-based reinforcement learning

1: Input: state sample procedure \( d \)
2: Input: model \( m \)
3: Input: policy \( \pi \)
4: Input: predictions \( v \)
5: Input: environment \( \mathcal{E} \)
6: Get initial state \( s \leftarrow \mathcal{E} \)
7: for iteration \( \in \{1, 2, \ldots, K\} \) do
8: \hspace{1cm} for interaction \( \in \{1, 2, \ldots, M\} \) do
9: \hspace{2cm} for planning step \( \in \{1, 2, \ldots, P\} \) do
10: \hspace{3cm} end for
11: \hspace{1cm} end for
12: end for

DQN is model-based due to its replay buffer

Interactions with the real Environment

Take advantage of the model

Van Hasselt et al. 2019
The fourth and final element of some reinforcement learning systems is a model of the environment. This is something that mimics the behavior of the environment, or more generally, that allows inferences to be made about how the environment will behave.
The fourth and final element of some reinforcement learning systems is a *model* of the environment. This is something that mimics the behavior of the environment, or more generally, that allows inferences to be made about how the environment will behave.

Again, DQN’s replay buffer fits this description.
Distinction to normal Control Problems
Distinction to normal Control Problems

Control Problems

Model Based RL
Why I want a model
Why I want a model – Sample efficiency

Save exploration

High computational cost

Create Samples with Model
Why I want a model – Planning

Learn a model and then use a planning algorithm
Why I want a model – Transfer of Knowledge

Learn how to kick the ball

Concentrate on teamplay
The effect of Planning Shape On Dyna-style planning in High-dimensional State Spaces

How does Dyna perform at Arcade games in a Deep Learning Setting?

Zacharias Holland et al. 2018
Important parts

Model

Learner

Roll-out Shapes
Important parts

Model

Learner

Roll-out Shapes

Zacharias Holland et al. 2018
Train your Q-network
Train your Q-network
Replay Buffer

Planning Buffer

Train your Q-network

π

π M
Important parts

Model

Learner

Roll-out Shapes
Roll-out Shapes

100 x 1

Zacharias Holland et al. 2018
Roll-out Shapes

33 x 3

Zacharias Holland et al. 2018
Roll-out Shapes

10 x 10

Zacharias Holland et al. 2018
Experiments
Number of Samples & Benchmarks

(Rollout-)Dyna-DQN

DQN 100k

DQN Extra Updates

DQN 10M

Zacharias Holland et al. 2018
Relaxing the model

One model is pretrained on expert data

One model learns in an online fashion

Zacharias Holland et al. 2018
World Models

(Ha & Schmidhuber 2018, arXiv:1803.10122)
This thought bubble is our goal

Figure 1. A World Model, from Scott McCloud’s Understanding Comics. (McCloud, 1993; E, 2012)
Learner over all

Vision

Memory

Controller

Ha et al. 2018
Vision (V) Model

Original Observed Frame → Encoder → Z → Decoder → Reconstructed Frame

Ha et al. 2018
Memory (M) Model

Mixture Density Network

Ha et al. 2018
Controller (C) Model

\[ a_t = W_c \begin{bmatrix} z_t \\ h_t \end{bmatrix} + b_c \]
Experiments
Experiment 1 – Training the model
Experiment 1 – Training the model

(frame, action)
Experiment 1 – Training the model

[frame, action]
Experiment 1 – Training the model
Experiment 1 – Training the model
Experiment 1 – Training the model

(frame, action)

(z, action)
Experiment 1 – Training the Controller

\[ a_t = W_c \begin{bmatrix} z_t & h_t \end{bmatrix} + b_c \]

Evolution!

Ha et al. 2018
## Experiment 1

<table>
<thead>
<tr>
<th>Method</th>
<th>Avg. Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQN (<em>Prieur, 2017</em>)</td>
<td>343 ± 18</td>
</tr>
<tr>
<td>A3C (continuous) (<em>Jang et al., 2017</em>)</td>
<td>591 ± 45</td>
</tr>
<tr>
<td>A3C (discrete) (<em>Khan &amp; Elibol, 2016</em>)</td>
<td>652 ± 10</td>
</tr>
<tr>
<td>CEOBillionaire (Gym Leaderboard)</td>
<td>838 ± 11</td>
</tr>
<tr>
<td>V Model</td>
<td>632 ± 251</td>
</tr>
<tr>
<td>V Model with hidden layer</td>
<td>788 ± 141</td>
</tr>
<tr>
<td><strong>Full World Model</strong></td>
<td><strong>906 ± 21</strong></td>
</tr>
</tbody>
</table>

*Removed memory Model*

*Removed memory Model, but with hidden layer in Controller*
Experiment 2 – train in dream world
Experiment 2 – what about rewards?

Maximize survival time

Memory model predicts death

Train MDN-RNN (M) to model

\[ P(z_{t+1}, d_{t+1} \mid a_t, z_t, h_t). \]
Experiment 2 – Problems

Too low temperature  -> no shooting
Too high temperature  -> chaos

Controller exploits inaccuracies of the model

Ha et al. 2018
Experiment 2

<table>
<thead>
<tr>
<th>Temperature $\tau$</th>
<th>Virtual Score</th>
<th>Actual Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10</td>
<td>2086 ± 140</td>
<td>193 ± 58</td>
</tr>
<tr>
<td>0.50</td>
<td>2060 ± 277</td>
<td>196 ± 50</td>
</tr>
<tr>
<td>1.00</td>
<td>1145 ± 690</td>
<td>868 ± 511</td>
</tr>
<tr>
<td>1.15</td>
<td>918 ± 546</td>
<td>1092 ± 556</td>
</tr>
<tr>
<td>1.30</td>
<td>732 ± 269</td>
<td>753 ± 139</td>
</tr>
</tbody>
</table>

| Random Policy     | N/A           | 210 ± 108    |
| Gym Leader        | N/A           | 820 ± 58     |
World Models – Shortcoming

Latent representation $z$ optimized for reconstruction and not for task solving

Ha et al. 2018
See you on Piazza