Fitted Q Iteration

Firas Safadi
University of Liège
June 2011
Part I

Theory
Environment

* State described by
  \[ s = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \]

* Action defined by
  \[ a = (u_1, u_2, \ldots, u_m) \in \mathbb{R}^m \]

* Environment responds with reward and new state
  \[
  \begin{cases}
  r \in \mathbb{R} \\
  s' \in \mathbb{R}^n
  \end{cases}
  \]
* Maps state-action pairs to rewards

\[ R(s, a) \rightarrow r \]

* We need \( R \) to take good actions!
Maximizing reward over a horizon

* Assume a horizon

\[ T \in \mathbb{N} \]

* Maximum cumulated reward

\[ V_T(s) = \max_{a,a',...} \left( \frac{R(s,a) + R(s',a') + \cdots}{T} \right) \]

* Maximum cumulated reward for state-action pairs

\[ Q_{T+1}(s,a) = r + V_T(s') \]
Fitted Q idea

* Start with a set of samples $U_j(s, a, r, s')_j$

* Incrementally build $Q_T$ using supervised learning

Learn $Q_{i+1}$
Increase $i$

Use $Q_i$ to compute $V_i$

Use $V_i$ to compute $Q_{i+1}$
Fitted Q algorithm

* **Input**
  - $s$ set of states
  - $a$ set of actions
  - $r$ set of rewards
  - $s'$ set of next states
  - $K$ number of samples
  - $T$ horizon

* **Output**
  - $Q_T$

\[
Q(s, a) \leftarrow r
\]

for $i = 1$ to $T - 1$

\[
R_j = r_j + \max_{a'} Q(s', a')
\]

end

\[
Q(s, a) \leftarrow R
\]

end
Part II

Experiment
Problem

- **World**: 10 × 10 surface
- **State**: \((x, y, a, b)\) in \([0,10]^4\)
- **Action**: \((u, v)\) in \([-1,1]^2\)
- **Goal**: reach target (dist. \(\leq 1\))
- **Reward**: 1 if dist. \(\leq 1\), else 0

- Random initial position
- Random initial target position
- Only 10 moves available
Learning

* Feed-forward backpropagation neural network
* Approx. 16,000 samples
* Simulate 100 random actions to estimate optimum
Assessing the impact of the learning horizon on performance

- Learn for horizon = 1, 2, ..., 10 and play 1,000 games
- Repeat 10 times
- Total of 10,000 games per horizon
Results

The chart shows the win rate over different horizons. The x-axis represents the horizon, and the y-axis represents the win rate. The data points indicate a steady increase in win rate as the horizon increases.
Part III

Wrap-up
Advantages of fitted Q iteration

* Offline
* Model-free
* Works with random trajectories
Future work

- Try random forests and compare with neural networks performance
- Try different sampling methods
  - Generate samples around edges
  - Generate complete trajectories
  - Resampling
- Try on bigger problems with larger state space (i.e., MASH)
Part IV

Acknowledgments
References

* **Charles Desjardins**
  * Neural Fitted Q-Iteration (Martin Riedmiller, ECML 2005), 2007

* **Damien Ernst**
  * Computing near-optimal policies from trajectories by solving a sequence of standard supervised learning problems, 2006

* **Yin Shih**
Thanks

* Jean-Baptiste Hooock
  * Implementation (MASH)

* Nataliya Sokolovska
  * Testing

* Olivier Teytaud
  * Concepts (fitted Q iteration, benchmark)
The End

Thanks for listening!