Real-Time Machine Learning with Reinforcement Learning

Autonomous Systems Research

Engineering & Physical Sciences
Heriot-Watt University,
Edinburgh Centre For Robotics

Boris Mocialov
December, 2015
Outline

Machine Learning Paradigms

Real-Time Machine Learning

Reinforcement Learning

Real-Time Learning using Reinforcement Learning
Machine Learning Paradigms

- Unsupervised Learning
  \[ f(x) \]
- Supervised Learning
  \[ y = f(x) \]
- Reinforcement Learning
  \[ y = f(x) \]

Real-Time Learning

- Lifelong learning
- Observer pattern
- Near-optimal results
- Non-blocking learning
how-to Real-Time Learning

- Evolutionary Algorithms
- Reinforcement Learning
Evolutionary Algorithm

Initialisation: generate initial population \( p \) of random* solutions and evaluate their fitness using some \( f(p) \)

Repeat until some termination condition:

1. Choose best* \( p' \) from \( p \)
2. Apply generic operators (e.g. crossover, mutation) to \( p' \)  
   Result: new solutions \( r \). Evaluate their fitness using \( f(r) \)
3. Put some % of \( r \) into \( p \) using some update rule (e.g. substitute)

* - some variations are possible

Every 'some' is tunable
Initialisation: create agent

1. Lay out data
2. Let agent interact with it
3. Teach the agent some behaviour to interact with the data
Reinforcement Learning Components

- Agent (Action, Observation)
- Environment
  - Stochasticity
  - Observability
  - Continuity
  - Stationarity
- Reward (Immediate, Long-Term)
- Model
- Value Function
- Policy

\[ y = f(x) \]
Reinforcement Learning Challenges

- Exploration vs Exploitation
- Delayed Reinforcement
- Model Generation
Preliminaries: Model with Markov Decision Process (MDP)

- Set of states $S$
- Set of actions $A$
- Transition $T(S, A, S') \rightarrow P(S'|S, A)$
- Reward $R(S, A)$
Preliminaries: Task Types

- **Finite-Horizon**
  - Fixed number of iterations
  - Receding horizon

\[
V^{\pi}(S) = E\left[\sum_{t=0}^{h} R(S_t) | \pi, S_0 = S\right]
\]

- **Infinite-Horizon, Discounted**

\[
V^{\pi}(S) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t) | \pi, S_0 = S\right]
\]
Finding Value Function

\[ V^\pi(S) = E\left[ \sum_{t=0}^{\infty} \gamma^t R(S_t) \mid \pi, S_0 = S \right] \]

(Expected total payoff \((E)\) expansion & re-parenthesising \((\gamma)\) )

\[ \Downarrow \]

\[ V^\pi(S) = R(S) + \gamma \sum_{S'} P_{S\pi(S)}(S') \cdot V^\pi(S') \]

Bellman equation to solve \(V^\pi(S)\) for every state of MDP

In discrete case, \(n\) states, \(n\) number of \(V^\pi(S)\) equations
\[ \rightarrow \text{system of linear equations} \]
Finding Optimal Policy (Given Model)

- **Value Iteration**
  
  Initialisation: arbitrary $V^\pi(S)$ for every state
  
  Repeat until convergence:
  
  1. Update every $V^\pi(S)$ by taking the best available action

  $$V^*(S) = R(S) + \max_a \gamma \sum_{S'} P_{S\!a}(S') V^*(S')$$

- **Policy Iteration**

  Initialisation: guess initial policy
  
  Repeat until convergence:
  
  1. Evaluate policy by calculating $V^\pi(S)$ using Bellman’s equation
  2. Improve policy by taking the best available action

  $$\pi^*(S) = R(S) + \arg\max_a \gamma \sum_{S'} P_{S\!a}(S') V^*(S')$$
Finding Optimal Policy (Without Model)

- Generating Models
  - Certainty equivalence
  - Dyna
  - Queue-Dyna
  ...

- Not Generating Models
  - TD(\(\lambda\))
  - Q-Learning
  ...

Real-Time Learning using Reinforcement Learning

SOCS:
- No stochasticity
- Fully observable
- Discrete
- Non-Stationary

Set-Up:
- Q-Learning
- 200 Interactions
- Random Spawn
- Decaying Exploration
Real-Time Learning using Reinforcement Learning

Experiment Results - Video
Questions?
Bibliography


