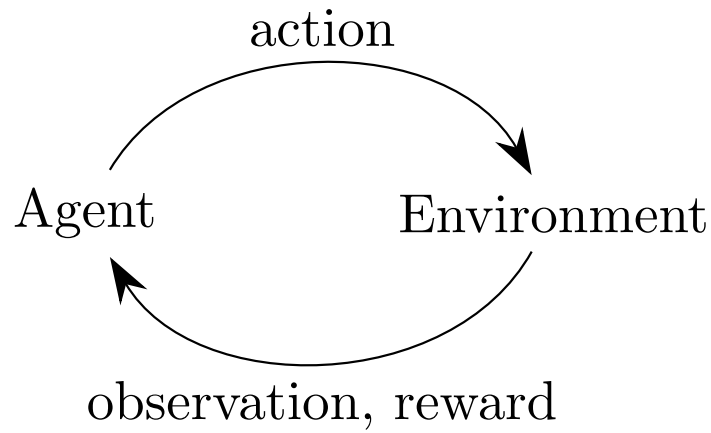
What is Reinforcement Learning?

- Branch of machine learning concerned with taking sequences of actions
- Usually described in terms of agent interacting with a previously unknown environment, trying to maximize cumulative reward



- Formalized as partially observable Markov decision process (POMDP)

Motor Control and Robotics

Robotics:

- Observations: camera images, joint angles
- Actions: joint torques
- Rewards: stay balanced, navigate to target locations, serve and protect humans

Business Operations

Inventory Management

- Observations: current inventory levels
- Actions: number of units of each item to purchase
- Rewards: profit

In Other ML Problems

- Classification with Hard Attention[1]
  - Observation: current image window
  - Action: where to look
  - Reward: $+1$ for correct classification
- Sequential/structured prediction, e.g., machine translation[2]
  - Observations: words in source language
  - Action: emit word in target language
  - Reward: sentence-level accuracy metric, e.g. BLEU score

---

[1] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. "Recurrent models of visual attention". *NIPS*. 2014.

[2] H. Daumé III, J. Langford, and D. Marcu. "Search-based structured prediction". (2009); S. Ross, G. J. Gordon, and D. Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning." *AISTATS*. 2011; M. Norouzi, S. Bengio, N. Jaitly, M. Schuster, Y. Wu, et al. "Reward augmented maximum likelihood for neural structured prediction". *NIPS*. 2016.

What Is Deep Reinforcement Learning?

Reinforcement learning using neural networks to approximate functions

- Policies (select next action)
- Value functions (measure goodness of states or state-action pairs)
- Dynamics Models (predict next states and rewards)

How Does RL Relate to Other ML Problems?

Supervised learning:

- Environment samples input-output pair $(x_t, y_t) \sim \rho$
- Agent predicts $\hat{y}_t = f(x_t)$
- Agent receives loss $\ell(y_t, \hat{y}_t)$
- *Environment asks agent a question, and then tells it the right answer*

How Does RL Relate to Other ML Problems?

Contextual bandits:

- Environment samples input $x_t \sim \rho$
- Agent takes action $\hat{y}_t = f(x_t)$
- Agent receives cost $c_t \sim P(c_t \mid x_t, \hat{y}_t)$ where P is an unknown probability distribution
- *Environment asks agent a question, and gives agent a noisy score on its answer*
- Application: personalized recommendations

How Does RL Relate to Other ML Problems?

Reinforcement learning:

- Environment samples input $x_t \sim P(x_t \mid x_{t-1}, y_{t-1})$
  - Environment is stateful: input depends on your previous actions!
- Agent takes action $\hat{y}_t = f(x_t)$
- Agent receives cost $c_t \sim P(c_t \mid x_t, \hat{y}_t)$ where P a probability distribution unknown to the agent.

How Does RL Relate to Other Machine Learning Problems?

Summary of differences between RL and supervised learning:

- You don't have full access to the function you're trying to optimize—must query it through interaction.
- Interacting with a *stateful* world: input x_t depend on your previous actions

1990s, beginnings

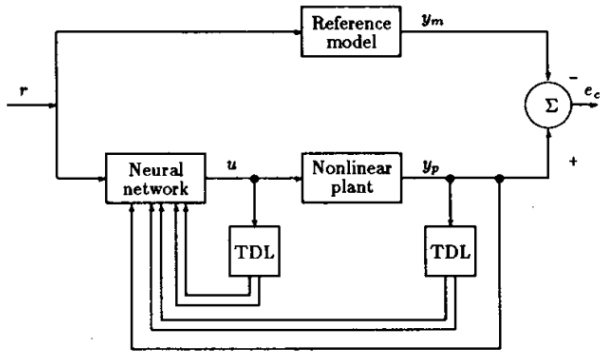
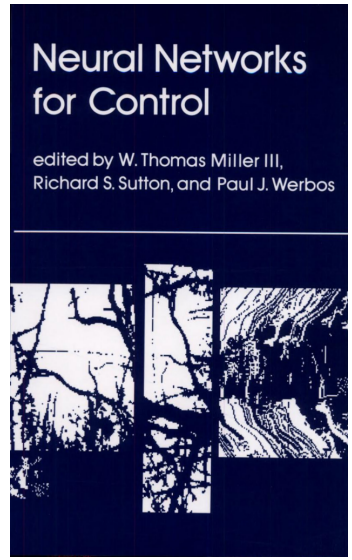


Fig. 21. Direct adaptive control of nonlinear plants using neural networks.



K. S. Narendra and K. Parthasarathy. "Identification and control of dynamical systems using neural networks". *IEEE Transactions on neural networks* (1990) W. T. Miller, P. J. Werbos, and R. S. Sutton. *Neural networks for control*. 1991

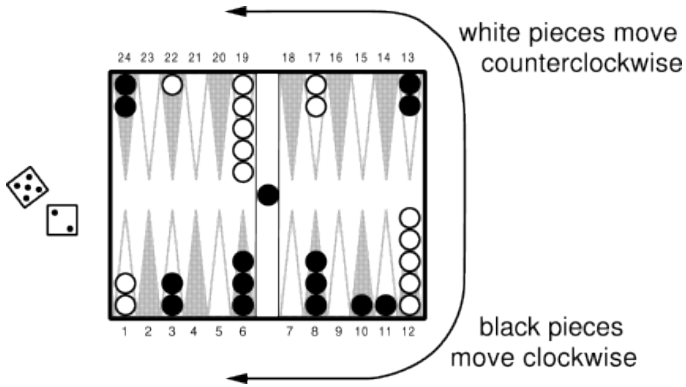
1990s, beginnings

This dissertation demonstrates how we can possibly overcome the slow learning problem and tackle non-Markovian environments, making reinforcement learning more practical for realistic robot tasks:

- Reinforcement learning can be naturally integrated with artificial neural networks to obtain high-quality generalization, resulting in a significant learning speedup. Neural networks are used in this dissertation, and they generalize effectively even in the presence of noise and a large number of binary and real-valued inputs.
- Reinforcement learning agents can save many learning trials by using an action model, which can be learned on-line. With a model, an agent can mentally experience the effects of its actions without actually executing them. Experience replay is a simple technique that implements this idea, and is shown to be effective in reducing the number of action executions required.
- Reinforcement learning agents can take advantage of instructive training instances provided by human teachers, resulting in a significant learning speedup. Teaching can also help learning agents avoid local optima during the search for optimal control. Simulation experiments indicate that even a small amount of teaching can save agents many learning trials.
- Reinforcement learning agents can significantly reduce learning time by hierarchical learning— they first solve elementary learning problems and then combine solutions to the elementary problems to solve a complex problem. Simulation experiments indicate that a robot with hierarchical learning can solve a complex problem, which otherwise is hardly solvable within a reasonable time.
- Reinforcement learning agents can deal with a wide range of non-Markovian environments by having a memory of their past. Three memory architectures are discussed. They work reasonably well for a variety of simple problems. One of them is also successfully applied to a nontrivial non-Markovian robot task.

L.-J. Lin. *Reinforcement learning for robots using neural networks*. Tech. rep. DTIC Document, 1993

1990s, beginnings



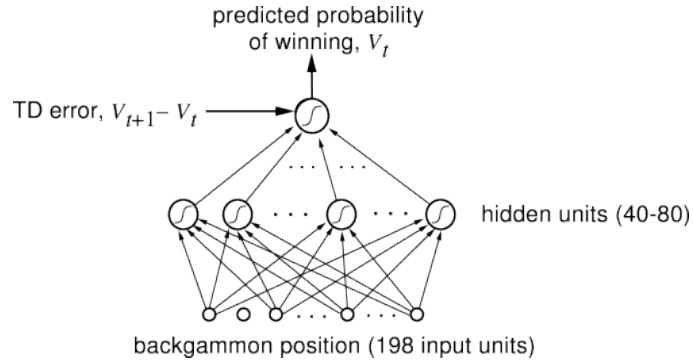


Table 11.1: Summary of TD-Gammon Results

Program	Hidden Units	Training Games	Opponents	Results
TD-Gam 0.0	40	300,000	other programs	tied for best
TD-Gam 1.0	80	300,000	Robertie, Magriel, ...	-13 pts / 51 games
TD-Gam 2.0	40	800,000	various Grandmasters	-7 pts / 38 games
TD-Gam 2.1	80	1,500,000	Robertie	-1 pt / 40 games
TD-Gam 3.0	80	1,500,000	Kazaros	$+6$ pts / 20 games

G. Tesauro. "Temporal difference learning and TD-Gammon". *Communications of the ACM* (1995). Figures from R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning*. MIT Press, 1998

Recent Success Stories in Deep RL

- ATARI using deep Q-learning[3], policy gradients[4], DAGGER[5]

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, et al. "Playing Atari with Deep Reinforcement Learning". (2013).

[4] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. "Trust Region Policy Optimization". (2015); V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, et al. "Asynchronous methods for deep reinforcement learning". (2016).

[5] X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang. "Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning". *NIPS*. 2014.

Recent Success Stories in Deep RL

- Robotic manipulation using guided policy search[6]

New Approach Trains Robots to Match Human Dexterity and Speed

By JOHN MARKOFF MAY 21, 2015

Guided by, from left, Chelsea Finn, Sergey Levine and Pieter Abbeel, Brett (the Berkeley Robot for the Elimination of Tedious Tasks) gets the peg into the hole. Peter Earl McCollough for The New York Times

RELATED COVERAGE

ROBOTICA EPISODE :
Prosthetic Limbs,
MAY 20, 2015

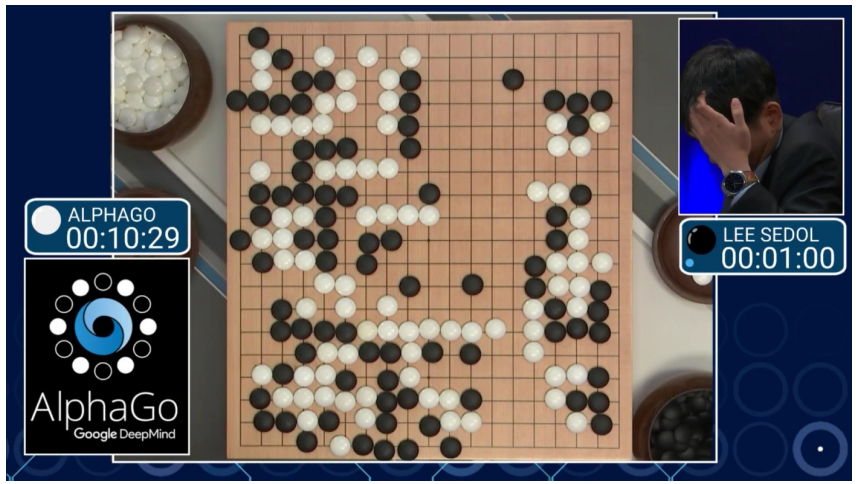
- Robotic locomotion using policy gradients[7]

[6] S. Levine, C. Finn, T. Darrell, and P. Abbeel. "End-to-end training of deep visuomotor policies". (2015).

[7] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. "High-dimensional continuous control using generalized advantage estimation". (2015).

Recent Success Stories in Deep RL

- AlphaGo: supervised learning + policy gradients + value functions + Monte-Carlo tree search[8]



[8] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, et al. "Mastering the game of Go with deep neural networks and tree search". *Nature* (2016).