CMSC 373 Artificial Intelligence Fall 2023 18-DeepQNetworks

Deepak Kumar
Bryn Mawr College

1

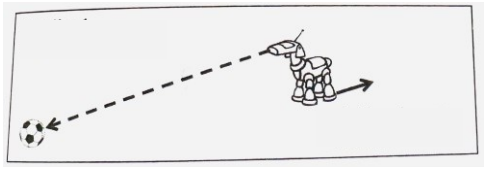
Agenda

- Reinforcement Learning
- Reinforcement Learning with Convolution Networks
- Monte Carlo Game Tree Search
- Success of Game Playing: Go

2

2

Reinforcement Learning



- Task

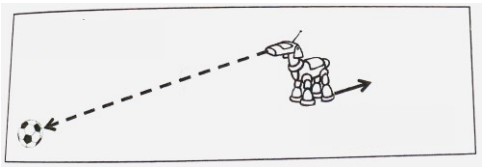
  A robot (*agent*) has to *find* and *kick* the ball.

- The agent is driven by a learning program that learns *without supervision*.
- The agent learns flexible strategies by performing actions. Occasionally it receives *rewards*.

3

3

Reinforcement Learning



- The agent learns flexible strategies by performing actions. Occasionally it receives *rewards*.

  Robots can perceive and estimate the distance to the ball.
  State: Robots estimate of distance to ball.

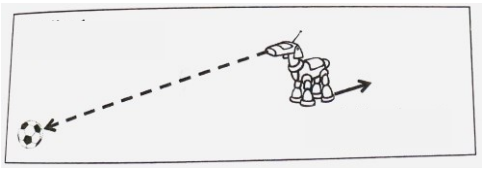
  Actions: Forward, Backward, Kick.

  Reward: 10 points if robot kicks and it hits the ball.

4

4

The Q Table



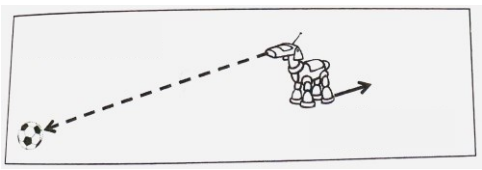
- **State**: Robots estimate of distance to ball.
- **Actions**: Forward, Backward, Kick.
- **Reward**: 10 points if robot kicks and it hits the ball.
- **Q Table**
  Keeps track of actions and their rewards at a given distance from ball.

State	0 steps away		1 step away		...	10 steps away		...
Action	Forward	0	Forward	0		Forward	0	
	Backward	0	Backward	0	...	Backward	0	...
	Kick	0	Kick	0		Kick	0	

5

5

The Q Table



- **Q Table**
  Keeps track of actions and their rewards at a given distance from ball.
- **Goal:** Learn values that are good predictions of upcoming rewards.
- **Algorithm**

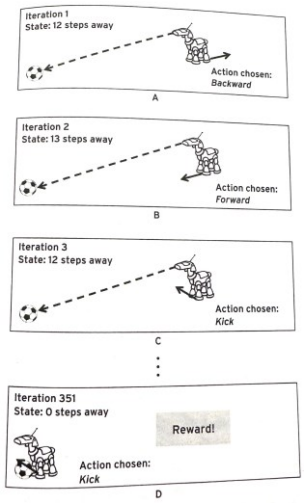
repeat
  Estimate the current state (x steps away)
  Look up state in Q Table (at entry x steps away)
  Use values in Q Table to choose an action (random if all values are 0)
  Perform the action
  Update reward if any (e.g. 10 points if kicked)

State	0 steps away		1 step away		...	10 steps away		...
Action	Forward	0	Forward	0		Forward	0	
	Backward	0	Backward	0	...	Backward	0	...
	Kick	0	Kick	0		Kick	0	

6

6

Episode 1

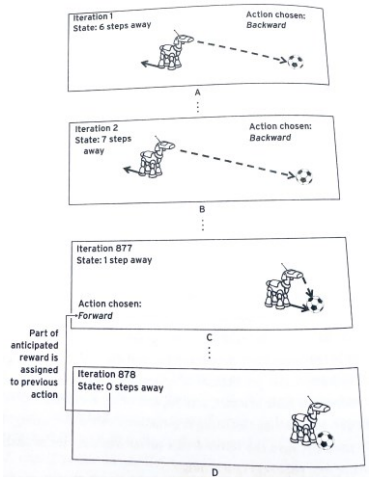


State	0 steps away		1 step away		...	10 steps away		...
Action	Forward	0	Forward	0		Forward	0	
	Backward	0	Backward	0	...	Backward	0	...
	Kick	**10**	Kick	0		Kick	0	

7

7

Episode 2



State	0 steps away		1 step away		...	10 steps away		...
Action	Forward	0	**Forward**	**8**		Forward	0	
	Backward	0	Backward	0	...	Backward	0	...
	Kick	**10**	Kick			Kick	0	

8

8

Reinforcement Q-Learning

- Gradually updating values in the Q Table until the robot has learned to perform the task from any starting point.
- **Issues**

  In real-world tasks perception of state is uncertain (e.g. how many steps?)
  Estimate of distance is a rough estimate.
  Effects of performing an action could be uncertain>
  Should it always choose an action with the highest reward value?
  Explore versus exploit balance.
  How many learning episodes?
  How many iterations/episode?
  How to discount the reward?
  Etc.
  [Sounds like tuning hyperparameters???]

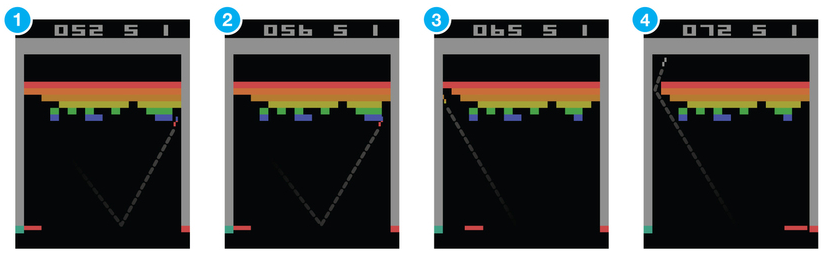
9

9

Game Playing with Reinforcement Learning

Atari Breakout

(play at: https://www.crazygames.com/game/atari-breakout)



From: Deep Mind, via https://nail.cs.ut.ee/index.php/2015/12/19/globular-star-cluster-radio-scope-great-turbulent-clouds/

Atari Learning Environment

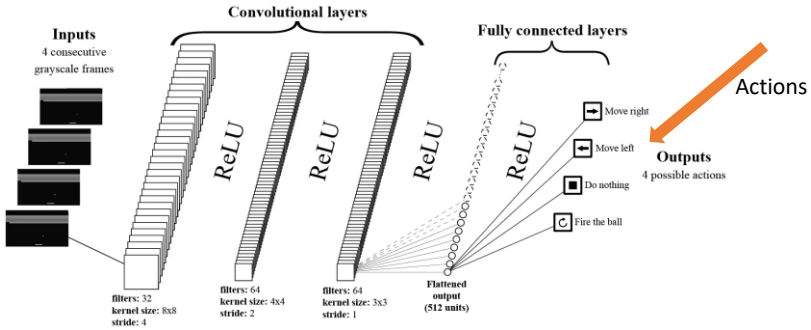
(https://www.endtoend.ai/envs/gym/atari/)

10

10

Q Learning with Convolution Networks

- **DQN (Deep Q-Network)**
  Replace the Q Table with a Convolution Network
  Instead of values in the Q Table, learn the weights.
  Learning adjusts the weights to minimize the difference between the current and previous iteration's outputs ("Learning a guess from a better guess").



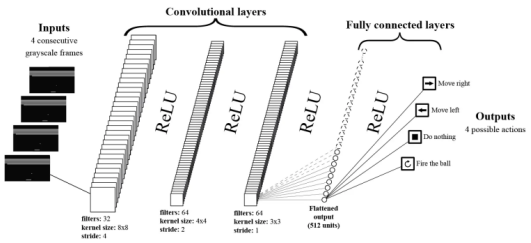
11

11

Q Learning with Convolution Networks

- **Deep Q-Learning**

Input current state.
The network outputs a value for each possible action (right, left, nothing, fire).
Choose and perform an action.
Input the new state to the network.
The network outputs a value for each possible action (right, left, nothing, fire).
Loss is the difference between previous state's outputs and the current state's outputs.
Use the loss to change the weights.



12

12

Atari Games with DQN (2013)

- DeepMind used DQN to 49 different games in the Atari Learning Environment.
- Each game had to be learned by a separate DQN.
- Took thousands of episodes to learn.
- DeepMind's DQN networks learned to play the games better than humans.

  On half of the 49 games the DQN was twice as good as humans.
  The other half the DQN was more than five times better!

13

13

DQN for 2-Person Games

- Classic game playing algorithms used Minimax to play Chess, Checkers, etc.
- Go, a more complex game, with a branching factor of over 250 is still challenging to play using Minimax.
- DeepMind applied DQN to learn to play Go.
- In 2016, **AlhpaGo Lee** beat World Champion Lee Sedol in a five game match.
- **AlphaGo** used a combination of specialized Go knowledge, DQN, Monte Carlo Game Tree Search, and supervised learning.
- Newer version **AlphaGo Zero** starts out with no knowledge at all. AlphaGo Zero beat AlphaGo Lee in every single game.

14

14

Monte Carlo Game Tree Search (MCTS)

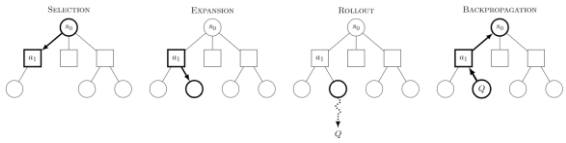
- **Monte Carlo Tree Search (MCTS)**

  Selection: Starting from current state select successive child nodes until a leaf node L is reached.

  Expansion: Unless L is an end game (win/loss/draw), expand L and choose one of them.

  Simulation: Complete one random playout (rollout) choosing randomly as it descends the tree.

  Backpropagation: Use result of rollout to update information on nodes in the path.
- AlphaGo Zero performs ~2000 rollouts/turn.



From: https://en.wikipedia.org/wiki/Monte_Carlo_tree_search#/media/File:MCTS_Algorithm.png

15

15

AlphaGo Zero: MCTS+DQN

- MCTS made a big improvement in the Go game player. But these programs were still not as good as humans.
- AlphaGo Zero: Combines MCTS with a deep convolution network.

  The CNN assigns (as before) a rough value to all possible moves from a game state.

  MCTS uses these values to kick-start its search.

  Results from the MCTS rollouts are then used to train the network. Essentially, the loss function uses the rollout's probabilities and compares them to the network's initial outputs to optimize the network's weights.

16

16

17

17

Implications of AlphaGo's Success

- Once again, false claims and hype followed.
- AlphaGo is a game self-learning architecture/algorithm. It is **not a general game playing engine**.
- Each AlphaGo game is a separate system, with its own separately trained CNN.
- **Transfer Learning:** Ability of a program to transfer its learning to another, completely different task. Humans are very good at transfer learning. Neural Networks are not. Transfer learning is an active area of research in Machine Learning.

DeepMind's AlphaGo mastered chess in its spare time

BY ERIC DAVID

AlphaGo has come a long way since it became the first artificial intelligence to conquer the game of Go nearly two years ago, but creator DeepMind Technologies, Google LLC's AI company, does not want AlphaGo to remain a one-trick pony.

It looks like it won't be. DeepMind revealed in a newly published research paper Tuesday that the latest version of AlphaGo has quickly mastered the games of chess and shogi using an algorithm that could mark another major step forward in AI development.

From: https://siliconangle.com/2017/12/06/deepminds-alphago-mastered-chess-spare-time/

18

18

Vocabulary

Action
AlphaGo Zero
AlphaGo
Atari
Atari Learning Environment
Breakout
DeepMind
DQN – Deep Q-Network
Episode
Monte Carlo Tree Search
Q Learning
Q Table
Reward
State
Unsupervised Learning
Reinforcement Learning

19

19

References

- M. Mitchell: *Artificial Intelligence: A Guide For Thinking Humans*, Farrar, Strouss, Giroux, 2019.
- M. Wooldridge: *A Brief History of Artificial Intelligence*. Flatiron Books, 2020.
- *Monte Carlo Tree Search*. Wikipedia. https://en.wikipedia.o rg/wiki/Monte_Carlo_tree_search (11/2023)

20

20