Artificial Intelligence CS 534 Fall-2017

Reinforcement Learning and Searching in Game

2017/12/11

Qi Wang, Abudula Aihaitijiang, Zeling Lei

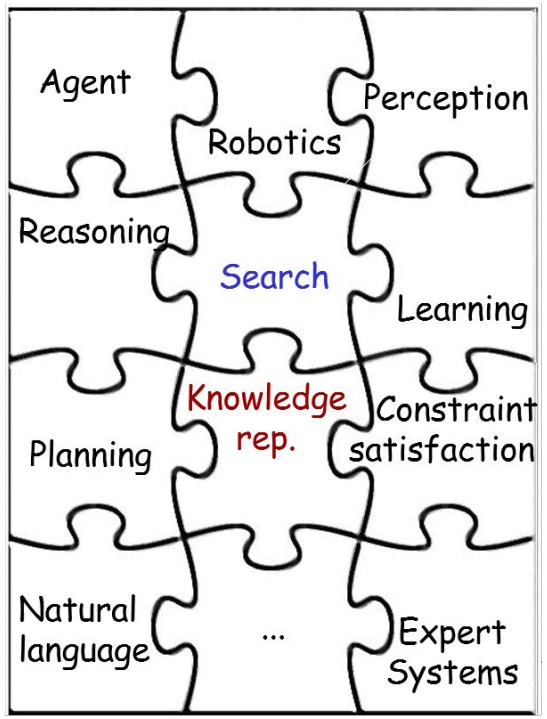
Introduction & Motivation

Artificial Intelligence

- Now AI technology has become far more effective and widely available
- AI is the study of ideas that enable computers to be intelligent
- Main areas of AI
- Video game

Motivation

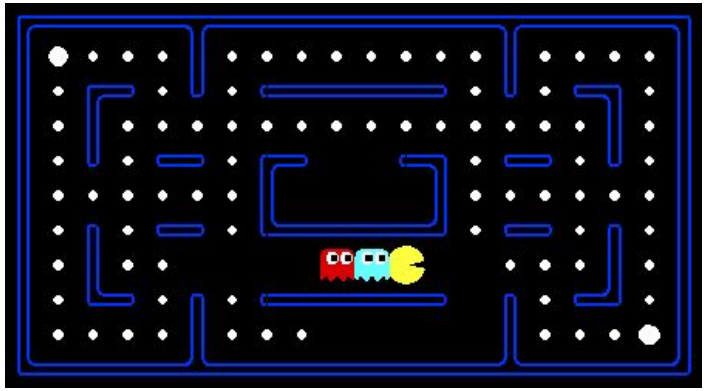
- In video games, Pac-man constitutes a very interested test environment (for learning AI techniques)
- Complex enough to be challenging
- Implement learning algorithms in the Pac-man game and visualize the results
- Better understanding foundational AI concepts



Background

- The Pac-man Game
  - Pac-man developed by Namco 1980 (Japan). Simple maze game
  - Ghosts played by the computer and chase the Pac-man and try to catch it
  - Pac-man wins if it has eaten all dots, loses if caught by a Ghost
  - Pac-Man can move (at most) up, down, left and right

- Learning Algorithms
  - Reinforcement learning methods are widely used in computer games
  - A method to generate policies for agent tasked with making decisions.
  - Typically represented by a Markov Decision Process



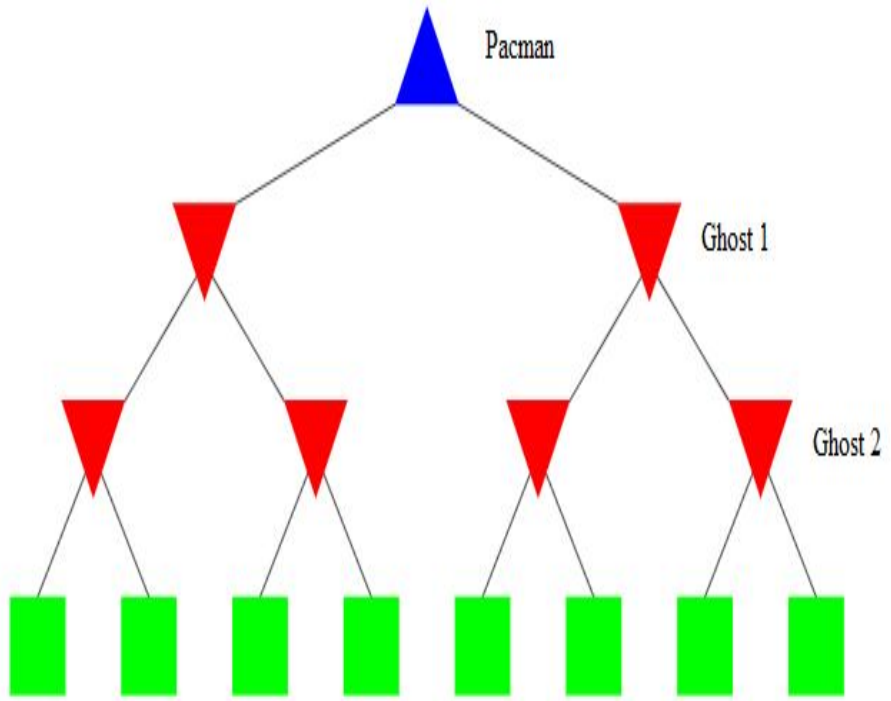
Method and Technical Approach

We will discuss two approaches for programming intelligence, learnability and adaptability into the game:

- Adversarial Search algorithms
  - Minimax
  - Alpha-beta Pruning
  - Expectimax
- Reinforcement Learning
  - Q-Learning
  - Approximate Q-Learning

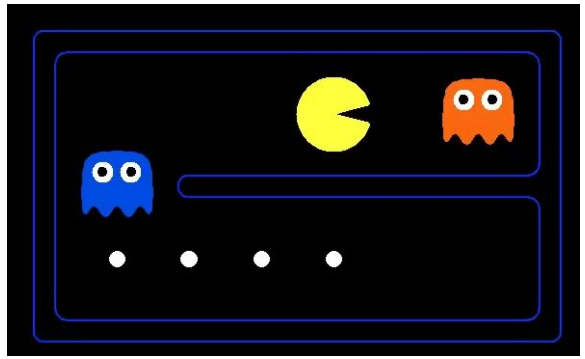
Method-Adversarial Search

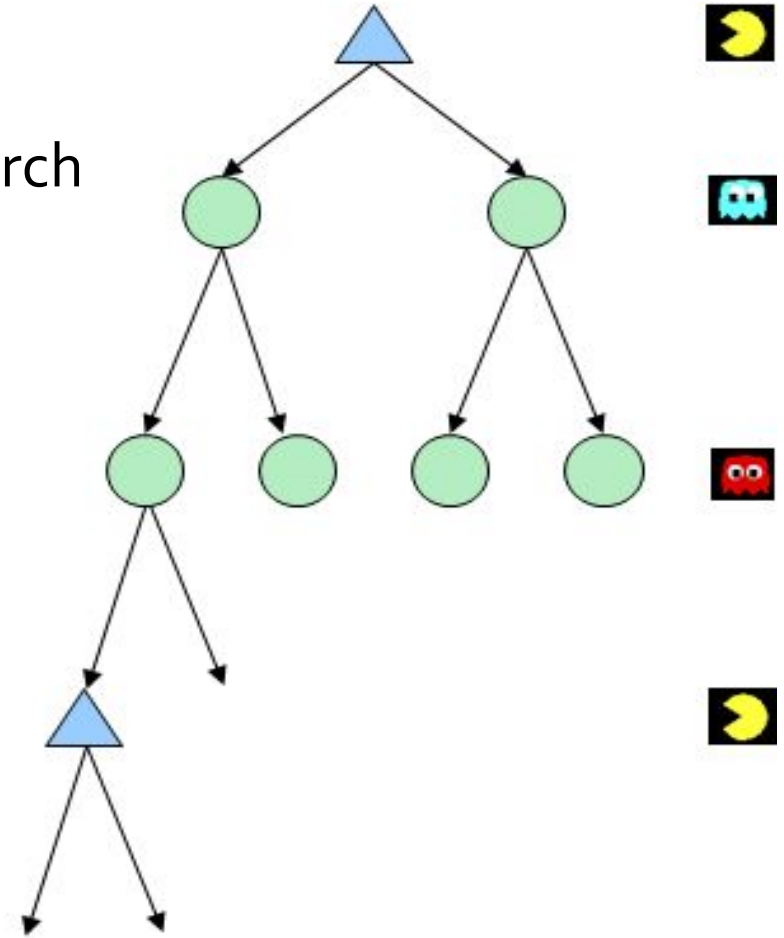
- Minimax Search
  - choose move with the highest minimax value
  - Propagate the utility values bottom -up using MIN and MAX operator
  - At the root node selects the move with the max utility value
- Minimax with Alpha-beta pruning
  - avoid searching subtrees of moves which won't be selected



Method-Adversarial Search

- Expectimax Search
  - Max nodes (Pacman) as in Minimax Search
  - Chance nodes (ghosts), like min nodes, except the outcome is uncertain
  - Take weight average (expectation) as expected utilities





Q-Learning

- $Q(s,a) \approx r + \gamma \max_{a'} Q(s',a')$
  - Q is the utility, r is the reward by taking action a, rmax(s',a') is the future reward by taking action a.

- Incorporate the new estimate into a running average

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + (\alpha)\left[r + \gamma \max_{a'} Q(s',a')\right]$$

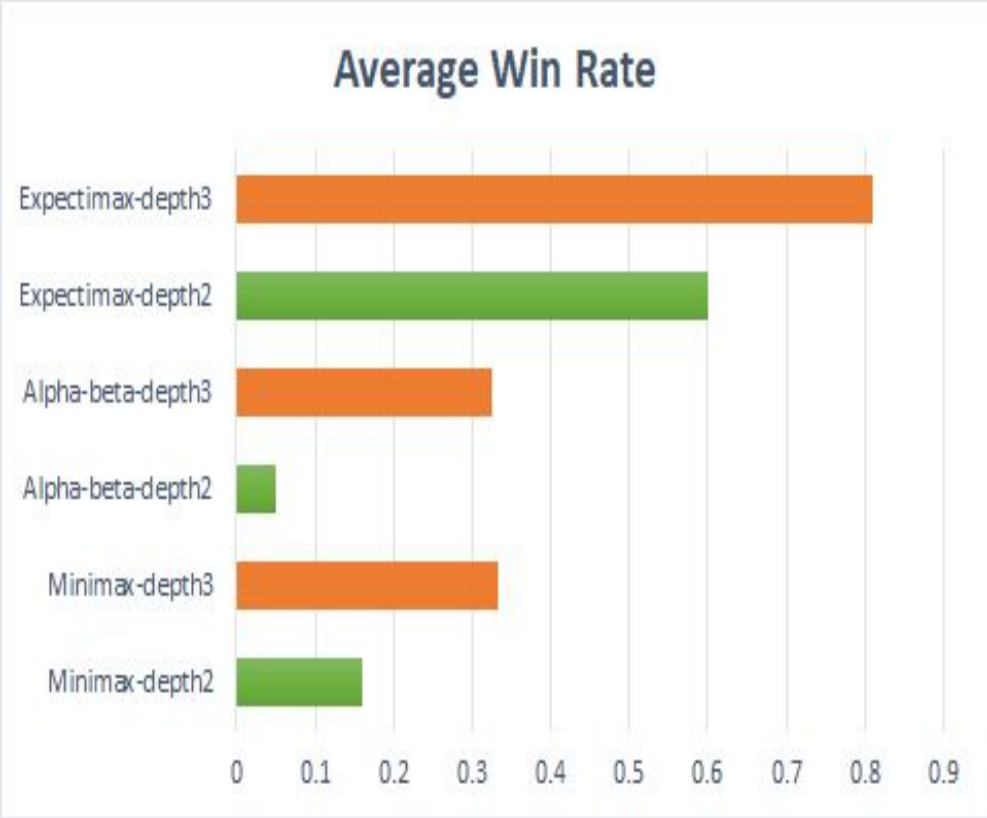
- However, basic Q-Learning keeps a table of all q-values. In reality, we cannot possibly learn about every single state.

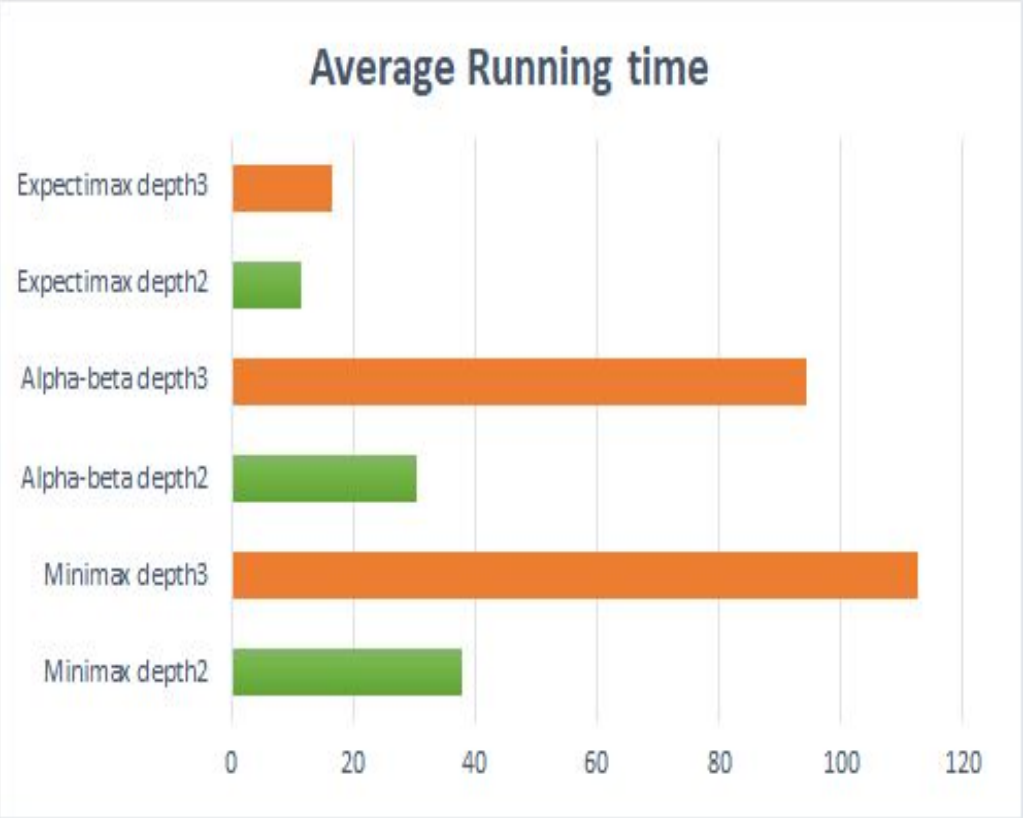
Approximate Q-Learning

- New method to calculate the Q value
  - $Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$
  - $\text{difference} = \left[r + \gamma \max_{a'} Q(s',a') \right] - Q(s,a)$
  - $w_i \leftarrow w_i + \alpha \, [\text{difference}] \, f_i(s,a)$

- Learn about some small number of training states from experience. Generalize that experience to new, similar situations.

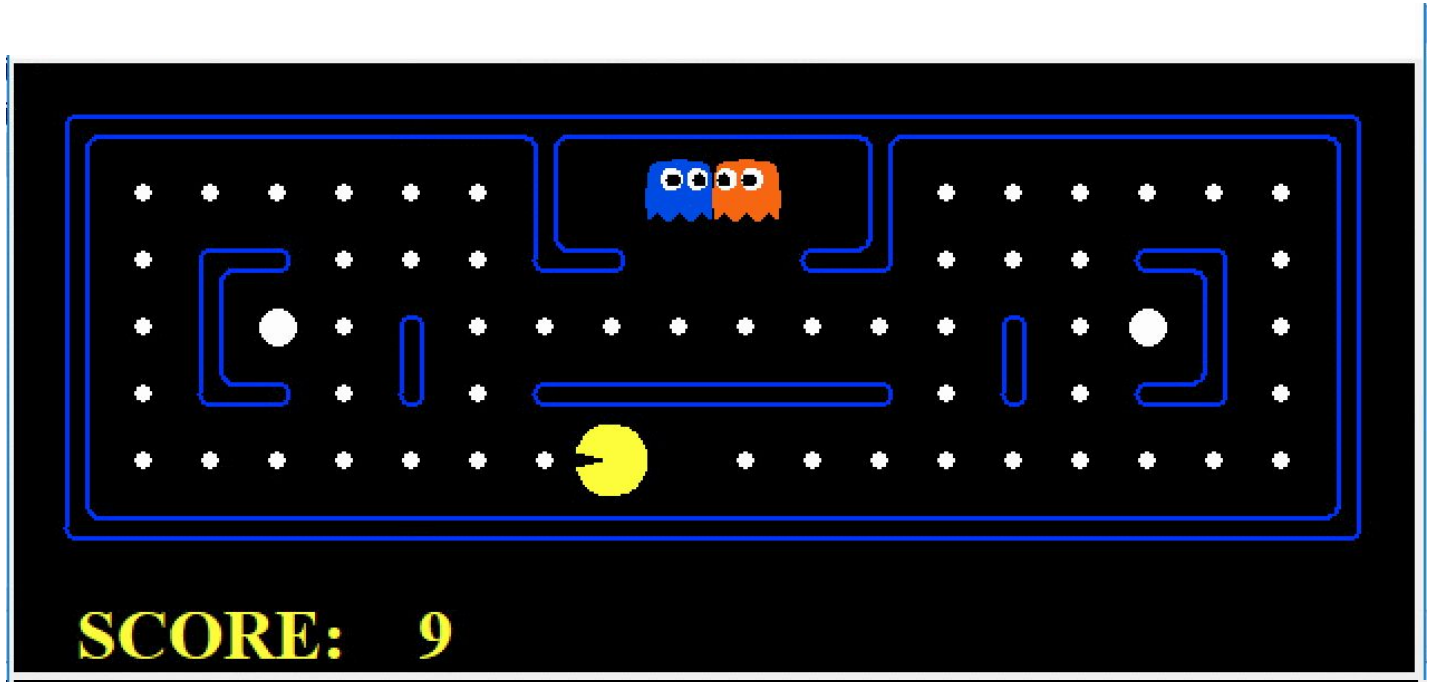
Results


Average Win Rate
Expectimax-depth3
Expectimax-depth2
Alpha-beta-depth3
Alpha-beta-depth2
Minimax-depth3
Minimax-depth2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

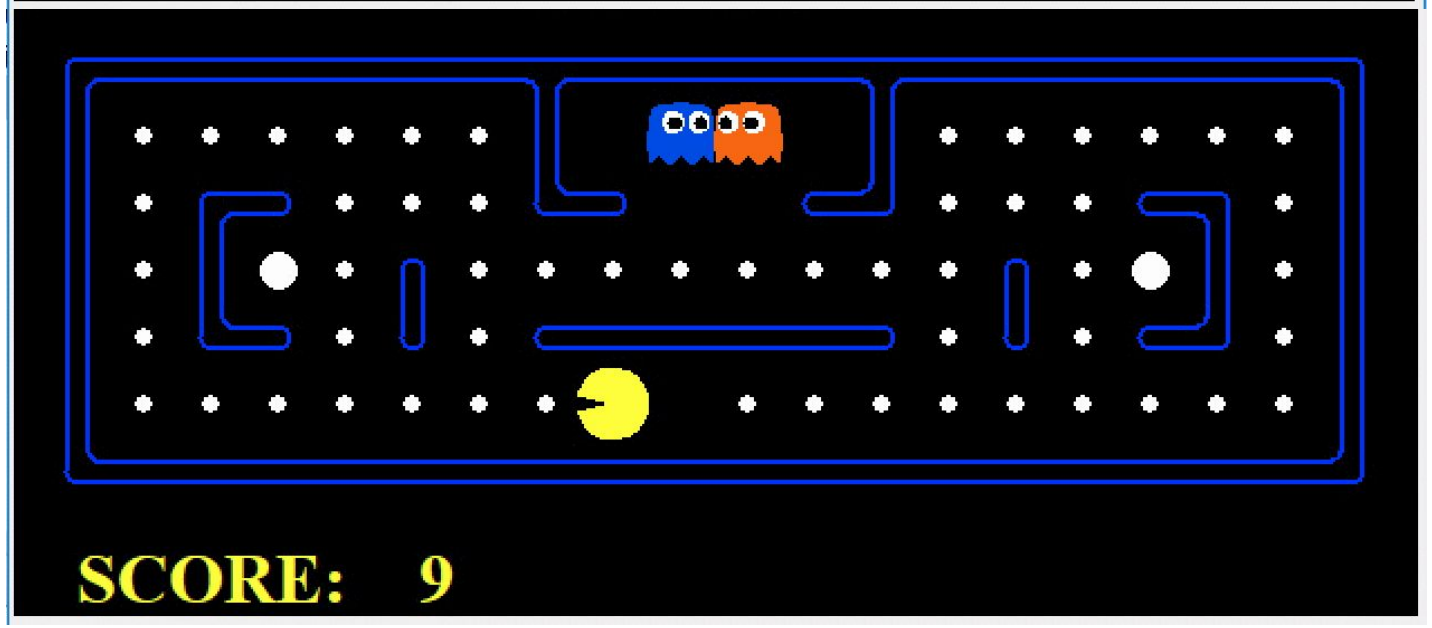

Average Running time
Expectimax depth3
Expectimax depth2
Alpha-beta depth3
Alpha-beta depth2
Minimax depth3
Minimax depth2
0
20
40
60
80
100
120

Results

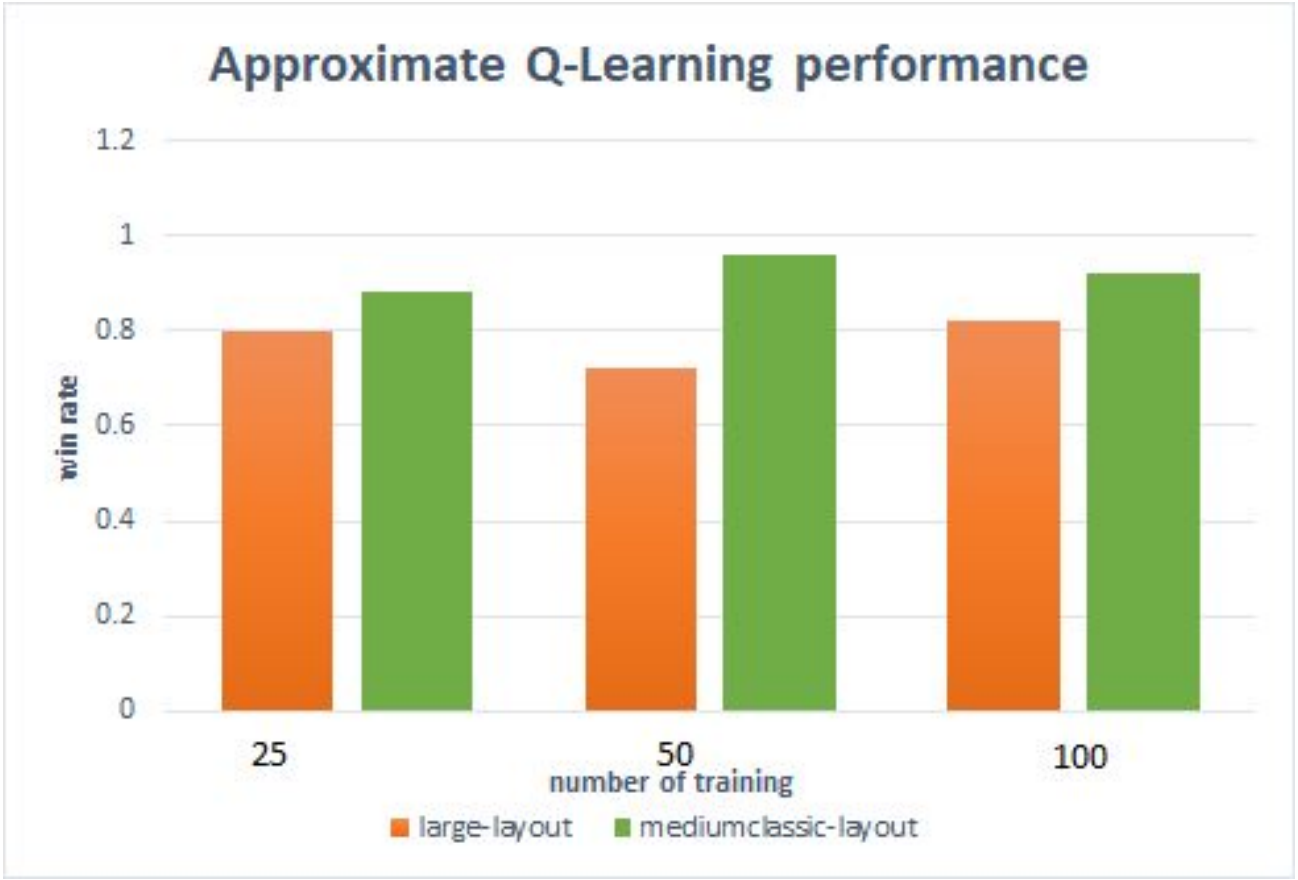
- Expectimax Search



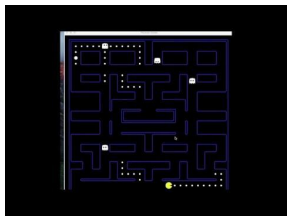
- Minimax Search



Results


Approximate Q-Learning performance
1.2
1
0.8
0.6
0.4
0.2
0
win rate
25
50
100
number of training
large-layout
mediumclassic-layout

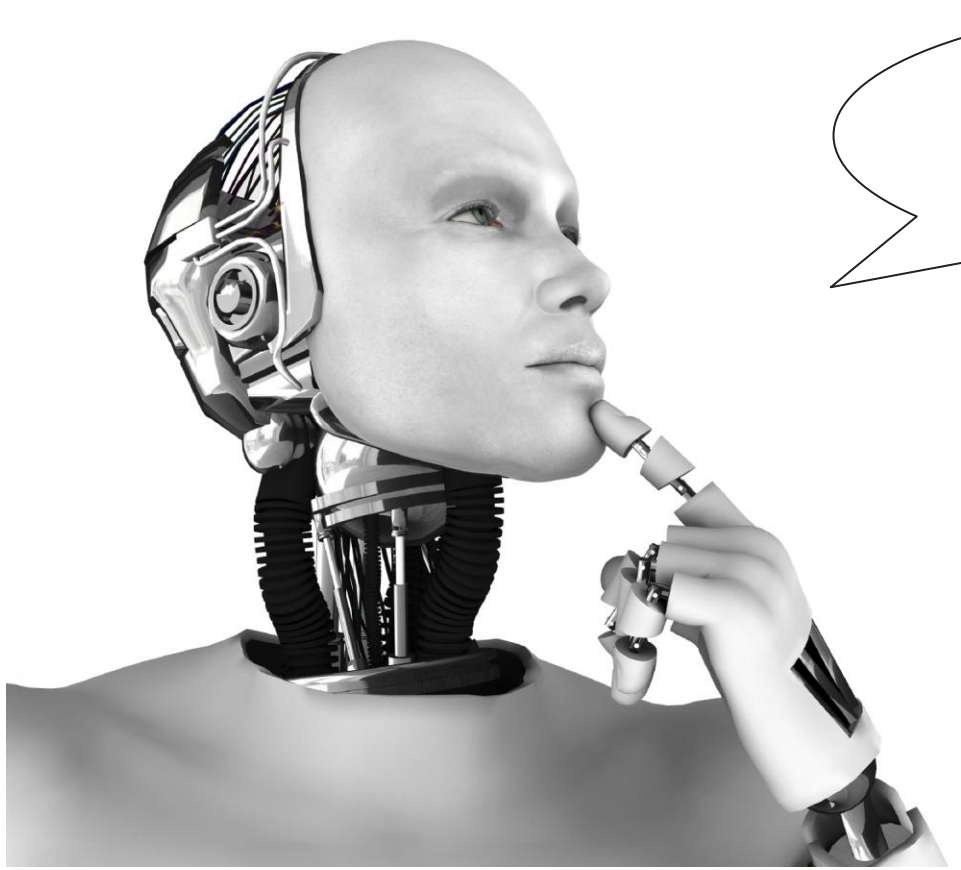

SCORE: 143



Conclusion

- Expectimax algorithm is much better than the minimax algorithm and alpha-beta pruning search. However, it has less foresight than Q-Learning and approximate Q-Learning.
- Pac-man using the approximate Q learning approach win 80 percent of the games for merely 25 times of training. But doesn't have a dramatic improvement of performance with more training.

Thank You !!!



Any Questions ?