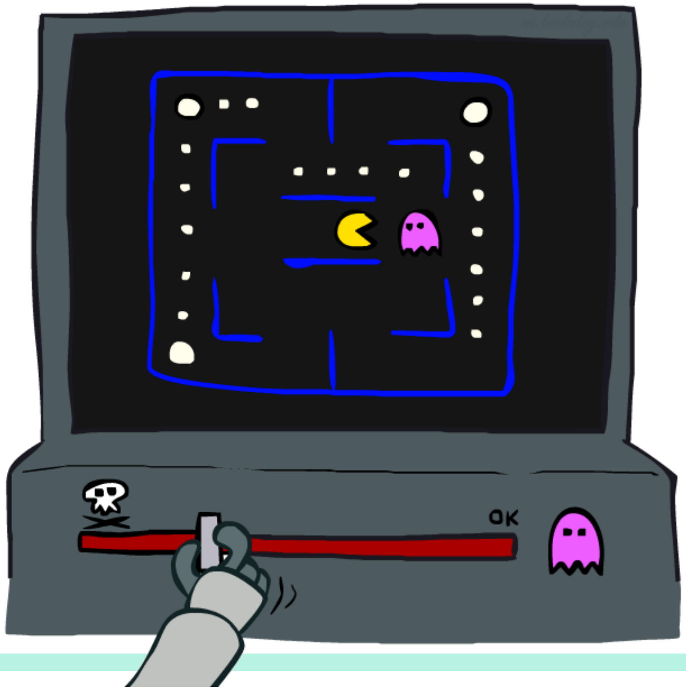
Advanced Topics in AI

Approximate Q-Learning



Instructor: Prof. Dr. techn. Wolfgang Nejdl

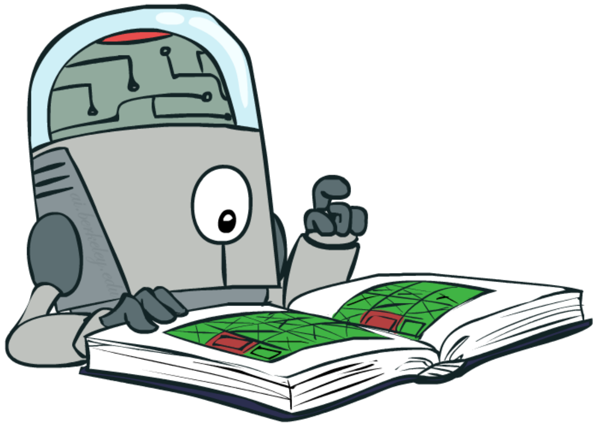
Leibniz University Hannover

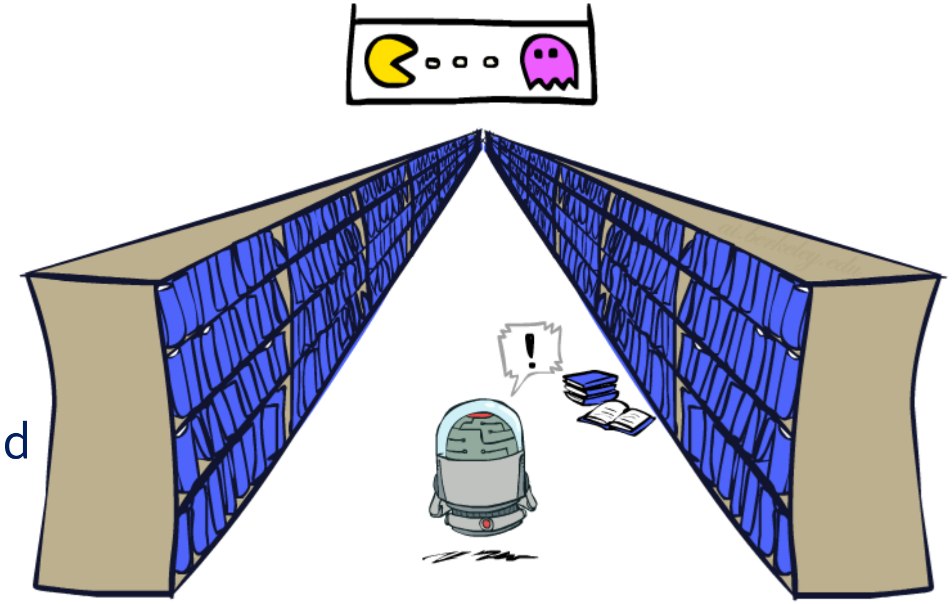
[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All materials are available at http://ai.berkeley.edu.]

Co-financed by the Connecting Europe Facility of the European Union

Generalizing Across States

- Basic Q-Learning keeps a table of all q-values
- In realistic situations, we cannot possibly learn about every single state!
  - Too many states to visit them all in training
  - Too many states to hold the q-tables in memory
- Instead, we want to generalize:
  - Learn about some small number of training states from experience
  - Generalize that experience to new, similar situations
  - This is a fundamental idea in machine learning, and we'll see it over and over again

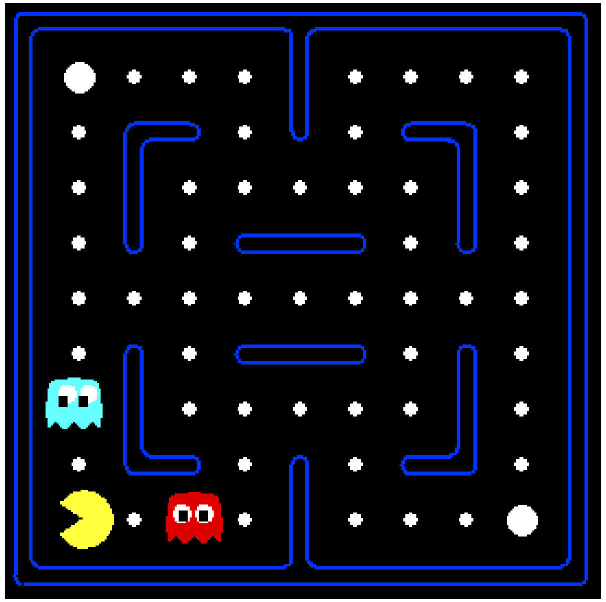




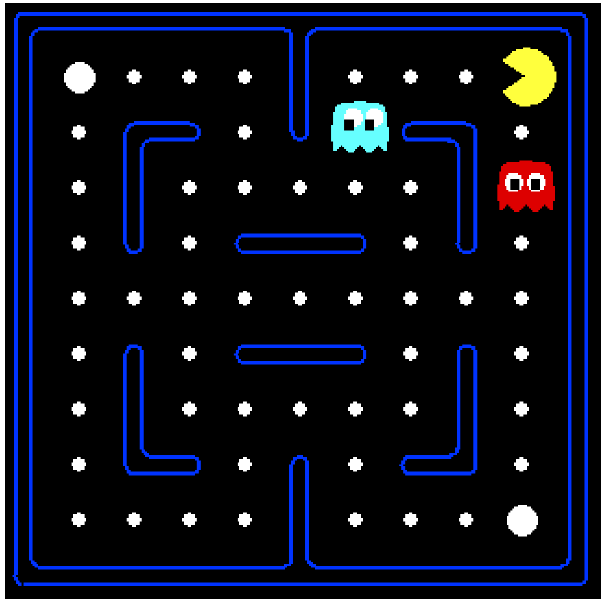
Co-financed by the Connecting Europe Facility of the European Union

Example: Pacman

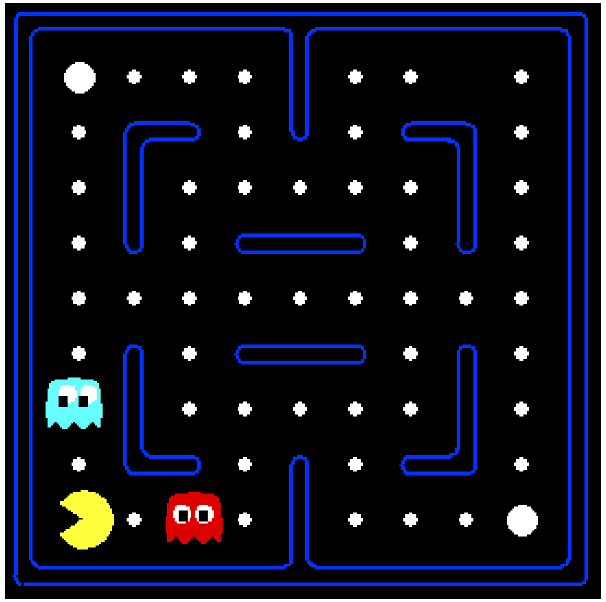
Let's say we discover through experience that this state is bad:



In naïve q-learning, we know nothing about this state:



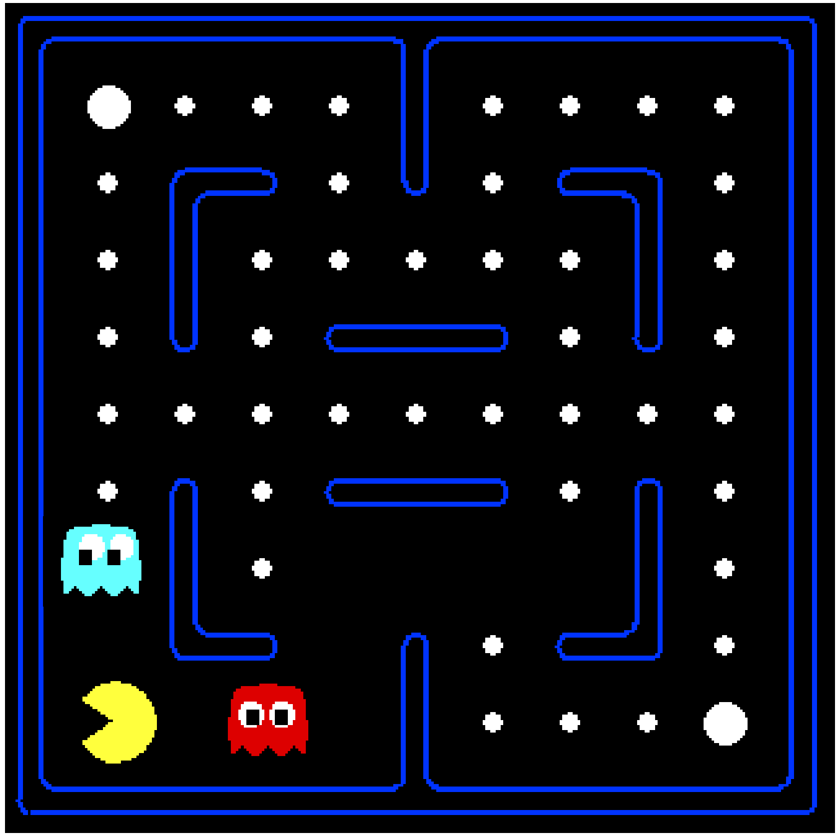
Or even this one!



Co-financed by the Connecting Europe Facility of the European Union

Feature-Based Representations

- Solution: describe a state using a vector of features (properties)
  - Features are functions from states to real numbers (often 0/1) that capture important properties of the state
  - Example features:
    - Distance to closest ghost
    - Distance to closest dot
    - Number of ghosts
    - 1 / (dist to dot)^2
    - Is Pacman in a tunnel? (0/1)
    - …… etc.
    - Is it the exact state on this slide?
  - Can also describe a q-state (s, a) with features (e.g. action moves closer to food)



Co-financed by the Connecting Europe Facility of the European Union

Linear Value Functions

- Using a feature representation, we can write a q function (or value function) for any state using a few weights:

$$V(s) = w_1 f_1(s) + w_2 f_2(s) + \ \dots \ + w_n f_n(s)$$

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ \dots \ + w_n f_n(s,a)$$

- Advantage: our experience is summed up in a few powerful numbers
- Disadvantage: states may share features but actually be very different in value!

Co-financed by the Connecting Europe Facility of the European Union

Approximate Q-Learning

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

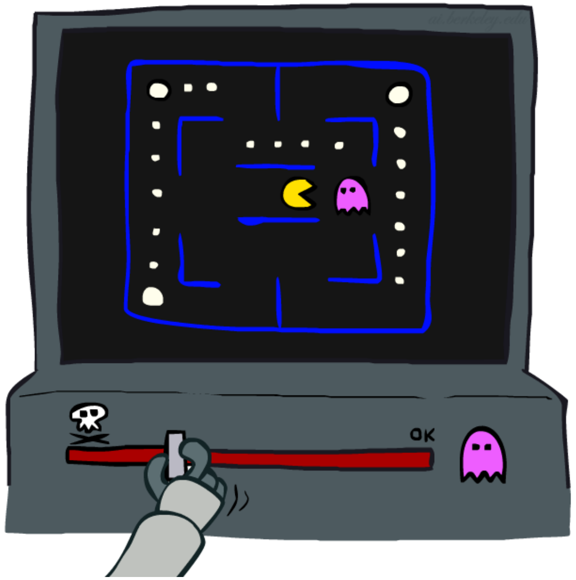
- Q-learning with linear Q-functions:

$$\text{transition} = (s,a,r,s'),\ \text{sample} = r + \gamma \max_{a'} Q(s',a')$$

$$\text{difference} = \left[r + \gamma \max_{a'} Q(s',a')\right] - Q(s,a)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \text{difference}$$ Exact Q's

$$w_i \leftarrow w_i + \alpha \cdot \text{difference} \cdot f_i(s,a)$$ Approximate Q's



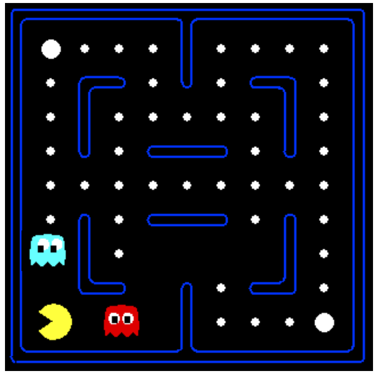
- Intuitive interpretation:
    - Adjust weights of active features
    - E.g., if something unexpectedly bad happens, blame the features that were on: dis-prefer all states with that state's features
- Formal justification: online least squares

Co-financed by the Connecting Europe Facility of the European Union

Example: Q-Pacman

$$Q(s,a) = 4.0 f_{DOT}(s,a) - 1.0 f_{GST}(s,a)$$

s



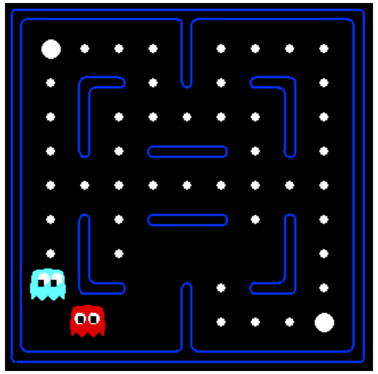
$$f_{DOT}(s, NORTH) = 0.5$$

$$f_{GST}(s, NORTH) = 1.0$$

$a = NORTH$

$r = -500$

s'



$$Q(s, NORTH) = +1$$

$$r + \gamma \max_{a'} Q(s', a') = -500 + 0$$

$$Q(s', \cdot) = 0$$

difference $= -501$

$$w_{DOT} \leftarrow 4.0 + \alpha \cdot (-501) \cdot 0.5$$

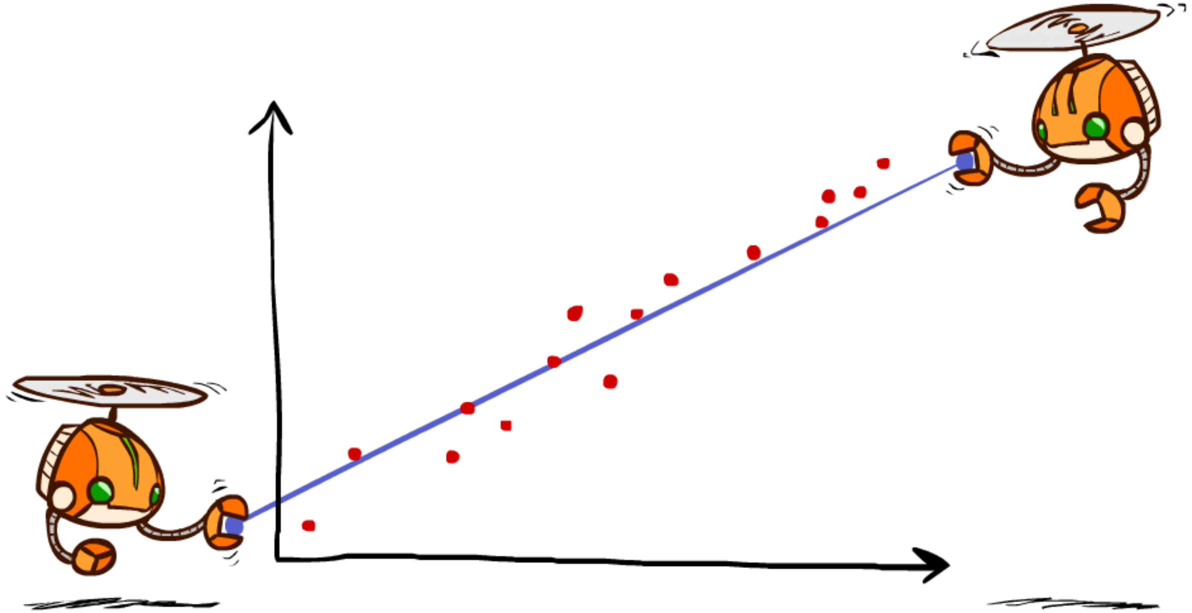
$$w_{GST} \leftarrow -1.0 + \alpha \cdot (-501) \cdot 1.0$$

$$Q(s,a) = 3.0 f_{DOT}(s,a) - 3.0 f_{GST}(s,a)$$

Co-financed by the Connecting Europe Facility of the European Union

Advanced Topics in AI

Next: Q-Learning and Least Squares



Instructor: Prof. Dr. techn. Wolfgang Nejdl

Leibniz University Hannover

[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All materials are available at http://ai.berkeley.edu.]