



Reinforcement Learning

Lecture 15: Learning through Approximation

10th Oct 2025

Instructor: Raunak Bhattacharyya



ScAI

YARDI SCHOOL OF ARTIFICIAL INTELLIGENCE
INDIAN INSTITUTE OF TECHNOLOGY DELHI

Recap



Seaquest, [Source](#)



Montezuma's Revenge, [Source](#)

'Real-World' RL

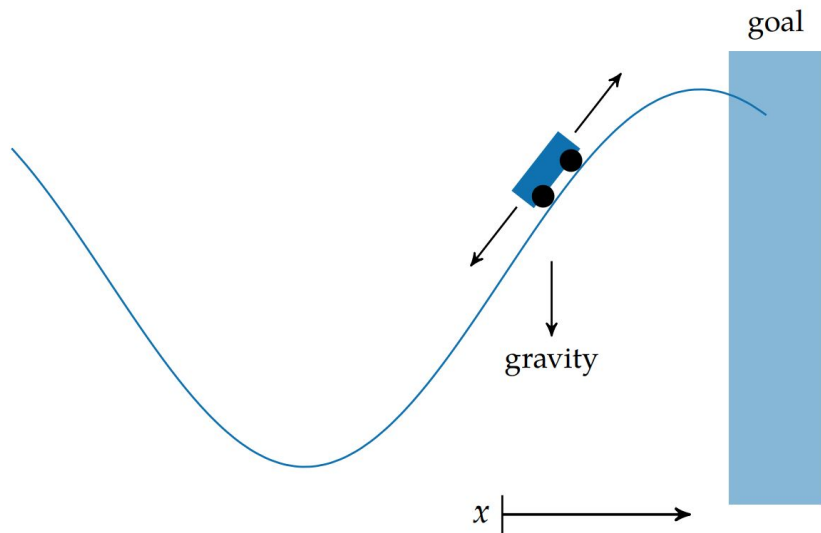


[Source: Deepmind, DQN](#)

128^{33600}



Mountain Car



[Source](#)

$$x \in [-1.2, 0.6]$$

$$v \in [-0.07, 0.07]$$

$$(a = -1)$$

$$(a = 1)$$

$$(a = 0)$$

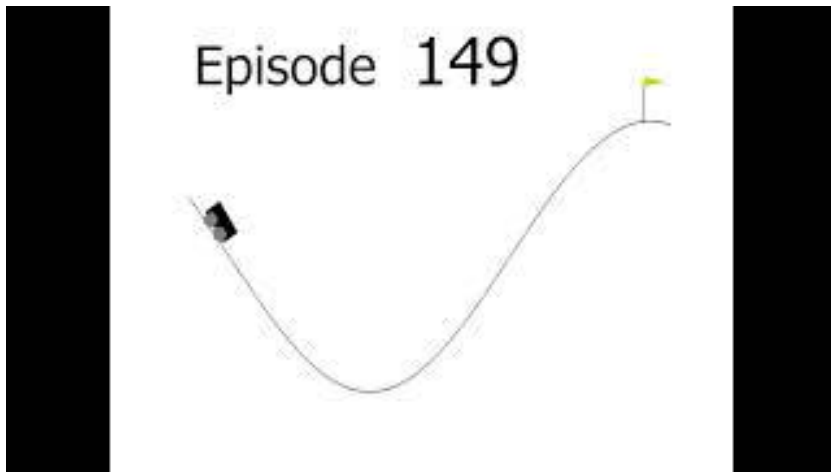
-1 reward every turn

past $x = 0.6$

Mountain Car

$$v' \leftarrow v + 0.001a - 0.0025 \cos(3x)$$

$$x' \leftarrow x + v'$$



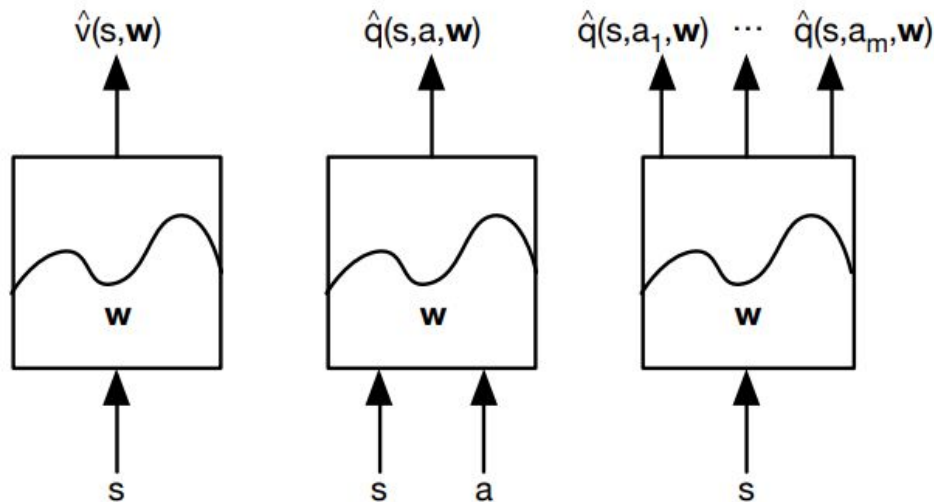
[Source](#)

Plan

- Approximating value functions
- VI w/ FA
- FA w/ Q-values

Problems with Large State Spaces

- Memory
- Time
- Need generalisation
- Function approximation



Example: Tetris

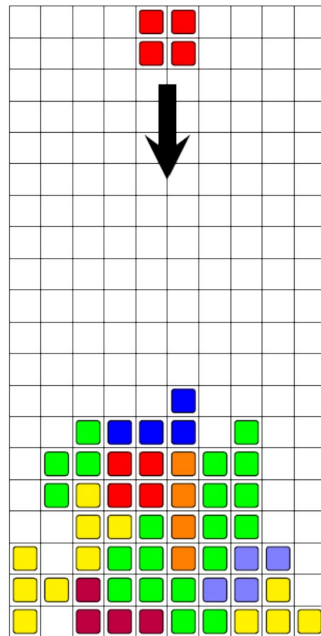
state: board configuration + shape of the falling piece $\sim 2^{200}$ states!

action: rotation and translation applied to the falling piece

22 features aka basis functions ϕ_i

- Ten basis functions, $0, \dots, 9$, mapping the state to the height $h[k]$ of each column.
- Nine basis functions, $10, \dots, 18$, each mapping the state to the absolute difference between heights of successive columns: $|h[k+1] - h[k]|$, $k = 1, \dots, 9$.
- One basis function, 19, that maps state to the maximum column height: $\max_k h[k]$
- One basis function, 20, that maps state to the number of 'holes' in the board.
- One basis function, 21, that is equal to 1 in every state.

$$\hat{V}_\theta(s) = \sum_{i=0}^{21} \theta_i \phi_i(s) = \theta^\top \phi(s)$$



Example: Pacman

$$\begin{aligned} &= \theta_0 \\ &+ \theta_1 \text{“distance to closest ghost”} \\ &+ \theta_2 \text{“distance to closest power pellet”} \\ &+ \theta_3 \text{“in dead-end”} \\ &+ \theta_4 \text{“closer to power pellet than ghost”} \\ &+ \dots \end{aligned}$$

$$= \sum_{i=0}^n \theta_i \phi_i(s) = \theta^\top \phi(s)$$

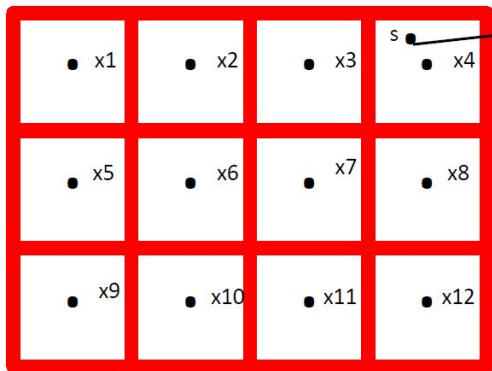


Example: 1 nearest neighbour

Only store values for $x_1, x_2, \dots, x_{12}$

– call these values $\theta_1, \theta_2, \dots, \theta_{12}$

Assign other states value of nearest “x” state

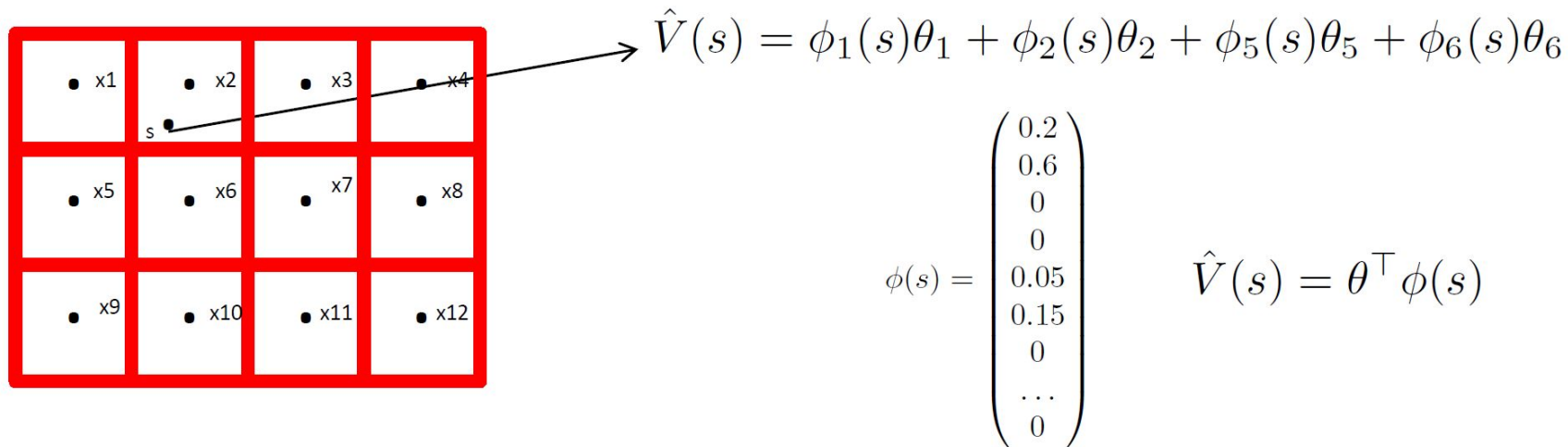


$$\hat{V}(s) = \hat{V}(x_4)$$

$$\phi(s) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ \dots \\ 0 \end{pmatrix}$$

$$\hat{V}(s) = \theta^\top \phi(s)$$

Example: 4 nearest neighbours



Further Examples

- $S = \mathbb{R}, \quad \hat{V}(s) = \theta_1 + \theta_2 s$
- $S = \mathbb{R}, \quad \hat{V}(s) = \theta_1 + \theta_2 s + \theta_3 s^2$
- $S = \mathbb{R}, \quad \hat{V}(s) = \sum_{i=0}^n \theta_i s^i$
- $S = \mathbb{R}^n \quad \hat{V}(s) = f_{\theta}(s) \quad (\text{e.g. neural net})$

Supervised Learning

Given:

- set of examples $(s^{(1)}, V(s^{(1)})), (s^{(2)}, V(s^{(2)})), \dots, (s^{(m)}, V(s^{(m)}))$,

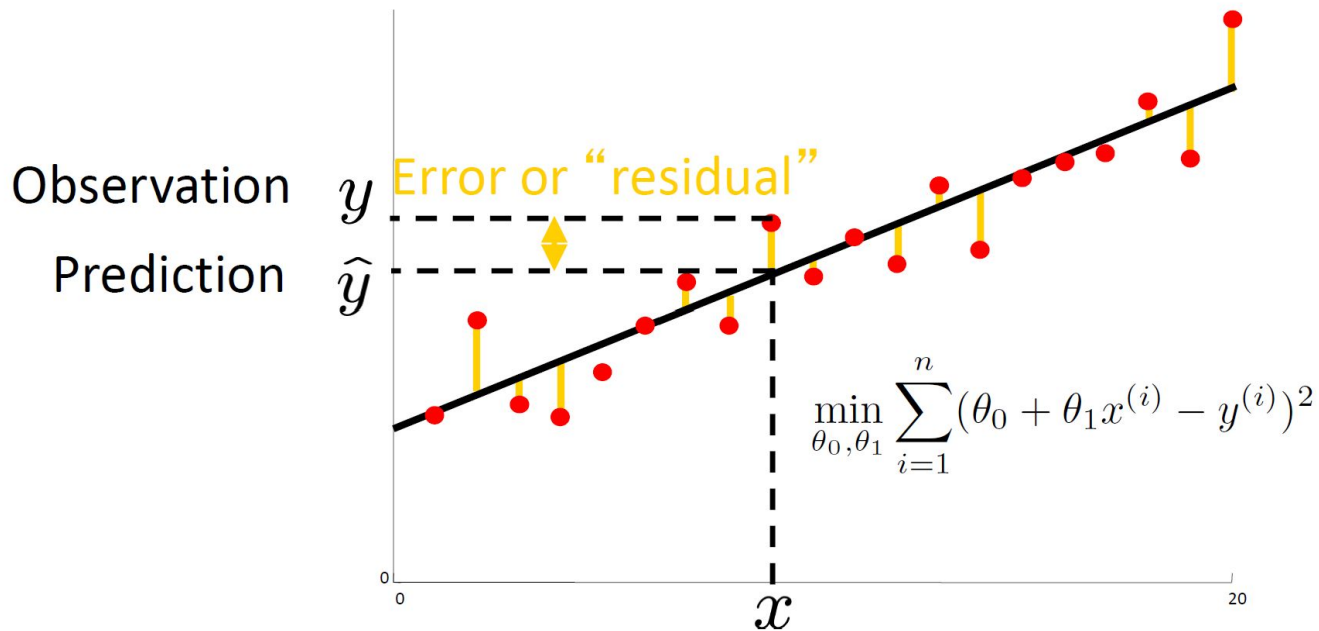
Asked for:

- “best” $\hat{V}_\theta$

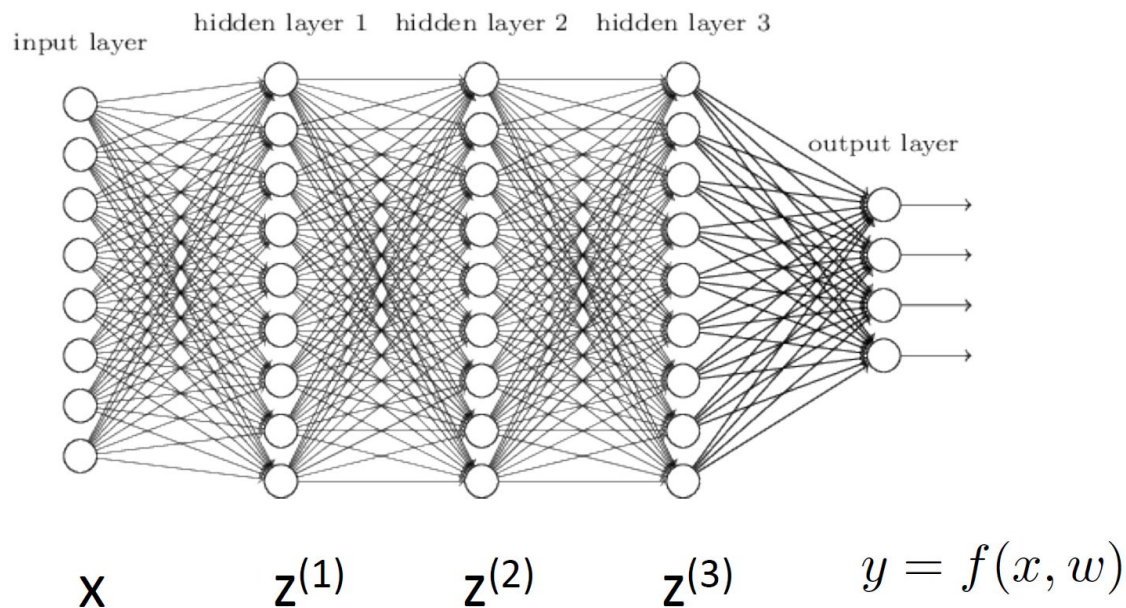
Representative approach: find θ through least squares

$$\min_{\theta \in \Theta} \sum_{i=1}^m (\hat{V}_\theta(s^{(i)}) - V(s^{(i)}))^2$$

Linear Regression

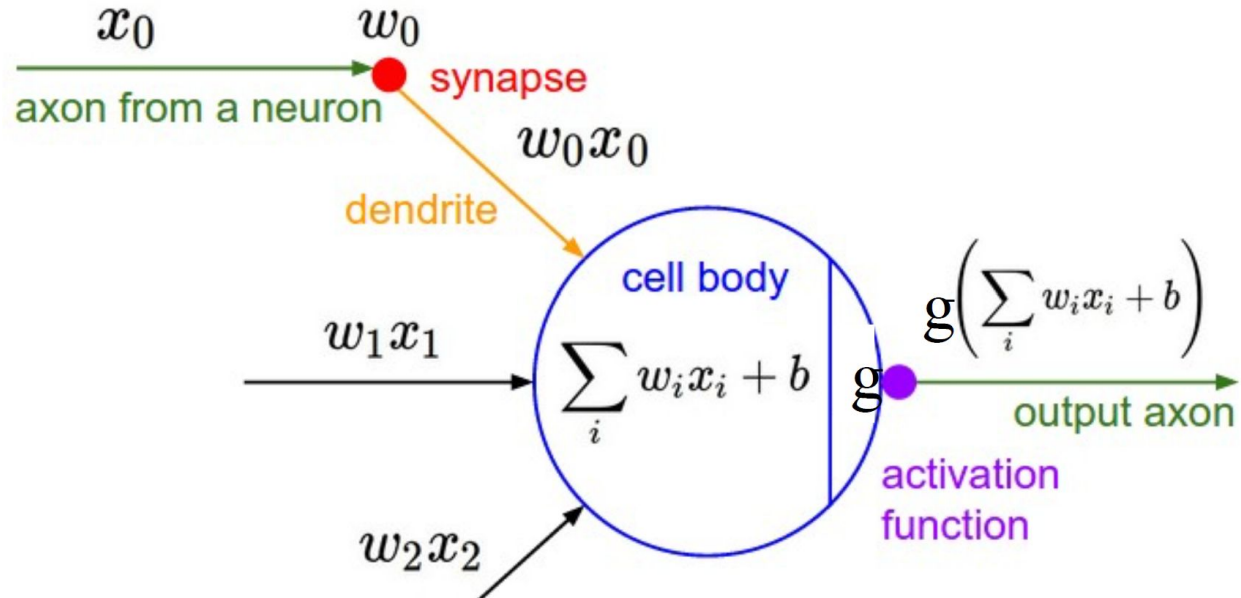


Neural Nets



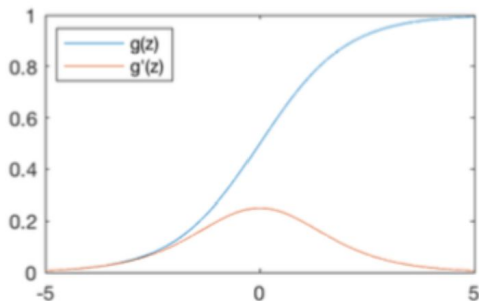
Choice of w determines the function from $x \rightarrow y$

Neural Nets



Neural Nets

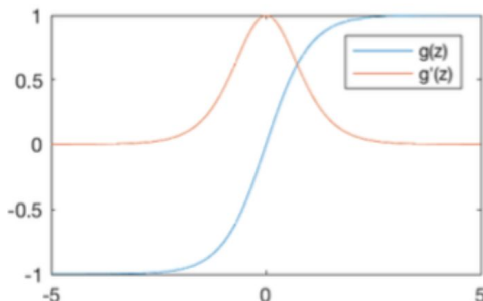
Sigmoid Function



$$g(z) = \frac{1}{1 + e^{-z}}$$

$$g'(z) = g(z)(1 - g(z))$$

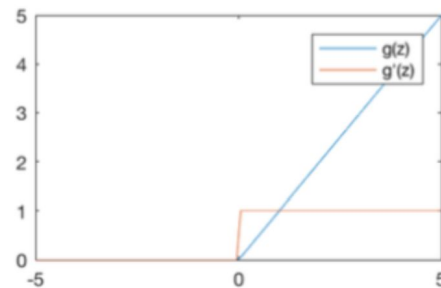
Hyperbolic Tangent



$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

$$g'(z) = 1 - g(z)^2$$

Rectified Linear Unit (ReLU)

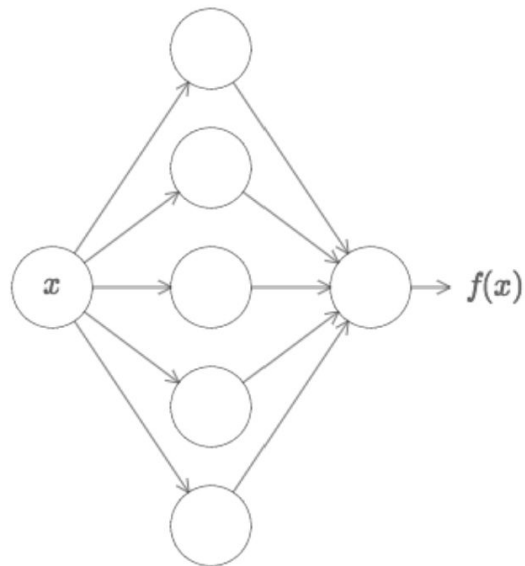
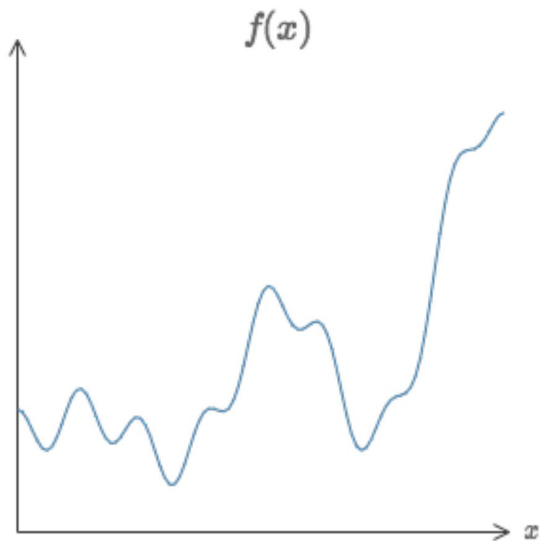


$$g(z) = \max(0, z)$$

$$g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases}$$

Neural Net Representation

Choice of w determines the function from $x \rightarrow y$

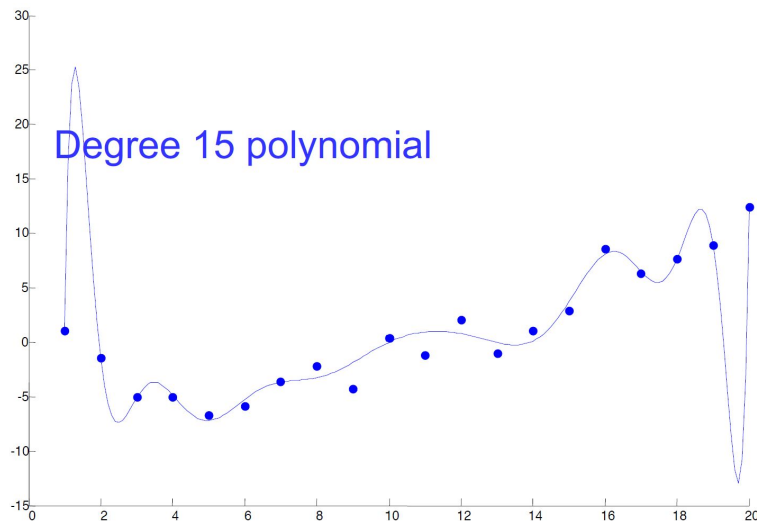


Universal Approximation Theorem

Hornik theorem 2: Whenever the activation function is *continuous, bounded and non-constant*, then, for arbitrary compact subsets $X \subseteq \mathbb{R}^k$, standard multilayer feedforward networks can approximate any continuous function on X arbitrarily well with respect to uniform distance, provided that sufficiently many hidden units are available.

Given any continuous function $f(x)$, if a 2-layer NN has enough hidden units, then there is a choice of weights that allow it to closely approximate $f(x)$.

Overfitting



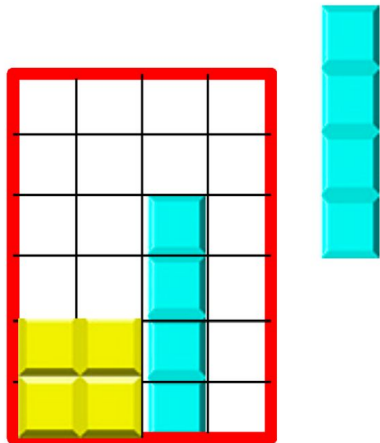
Do supervised learning. But where does the supervision come from?

VI with FA

Mini-Tetris: Setup

Mini-tetris: two types of blocks, can only choose translation (not rotation)

- Example state:



- Reward = 1 for placing a block
- Sink state / Game over is reached when block is placed such that part of it extends above the red rectangle
- If you have a complete row, it gets cleared

Mini-Tetris: State Subset and Features

$$S' = \left\{ \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} \begin{array}{c} \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \end{array} , \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} \begin{array}{c} \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \end{array} , \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} \begin{array}{c} \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \\ \text{yellow} \end{array} , \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} \begin{array}{c} \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \\ \text{cyan} \end{array} \right\}$$

- Four basis functions, $0, \dots, 3$, mapping the state to the height $h[k]$ of each of the four columns.
- Three basis functions, $4, \dots, 6$, each mapping the state to the absolute difference between heights of successive columns: $|h[k+1] - h[k]|$, $k = 1, \dots, 3$.
- One basis function, 7 , that maps state to the maximum column height: $\max_k h[k]$
- One basis function, 8 , that maps state to the number of 'holes' in the board.
- One basis function, 9 , that is equal to 1 in every state.

Init with $\vartheta^{(0)} = (-1, -1, -1, -1, -2, -2, -2, -3, -2, 10)$

Mini-Tetris: Bellman Backup

$$\begin{aligned}
 V(\text{Board 1}) = \max \{ & 0.5 * (1 + \gamma V(\text{Board 2})) + 0.5 * (1 + \gamma V(\text{Board 3})) , \\
 & 0.5 * (1 + \gamma V(\text{Board 4})) + 0.5 * (1 + \gamma V(\text{Board 5})) , \\
 & 0.5 * (1 + \gamma V(\text{Board 6})) + 0.5 * (1 + \gamma V(\text{Board 7})) , \\
 & 0.5 * (1 + \gamma V(\text{Board 8})) + 0.5 * (1 + \gamma V(\text{Board 9})) \}
 \end{aligned}$$

Mini-Tetris: Bellman Backup

$$\begin{aligned}
 V(\text{grid with I piece at (0,0)}) = \max \{ & 0.5 * (1 + \gamma V(\text{grid with I piece at (0,1)})) + 0.5 * (1 + \gamma V(\text{grid with I piece at (0,2)})), \\
 & 0.5 * (1 + \gamma V(\text{grid with I piece at (1,0)})) + 0.5 * (1 + \gamma V(\text{grid with I piece at (1,1)})), \\
 & 0.5 * (1 + \gamma V(\text{grid with I piece at (2,0)})) + 0.5 * (1 + \gamma V(\text{grid with I piece at (2,1)})), \\
 & 0.5 * (1 + \gamma V(\text{grid with I piece at (3,0)})) + 0.5 * (1 + \gamma V(\text{grid with I piece at (3,1)})) \}
 \end{aligned}$$

Mini-Tetris: Bellman Backup

$$\begin{aligned}
 V(\text{grid with I-piece}) = \max \{ & 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece and O-piece})), \\
 & \text{(6,2,4,0, 4, 2, 4, 6, 0, 1)} \qquad \qquad \qquad \text{(6,2,4,0, 4, 2, 4, 6, 0, 1)} \\
 & 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece and O-piece})), \\
 & \text{(2,6,4,0, 4, 2, 4, 6, 0, 1)} \qquad \qquad \qquad \text{(2,6,4,0, 4, 2, 4, 6, 0, 1)} \\
 & 0.5 * (1 + \gamma V(\text{grid with I-piece})) + 0.5 * (1 + \gamma V(\text{grid with I-piece and O-piece})), \\
 & \text{(sink-state, V=0)} \qquad \qquad \qquad \text{(sink-state, V=0)} \\
 & 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with I-piece and O-piece})) \} \\
 & \text{(0,0,2,2, 0,2,0, 2, 0, 1)} \qquad \qquad \qquad \text{(0,0,2,2, 0,2,0, 2, 0, 1)}
 \end{aligned}$$

Mini-Tetris: Bellman Backup

$$\begin{aligned}
 V(\text{board, piece}) = \max \{ & 0.5 * (1 + \gamma (\text{score} - 30)) + 0.5 * (1 + \gamma (\text{score} - 30)), \\
 & 0.5 * (1 + \gamma (\text{score} - 30)) + 0.5 * (1 + \gamma (\text{score} - 30)), \\
 & 0.5 * (1 + \gamma (\text{score} + 0)) + 0.5 * (1 + \gamma (\text{score} + 0)), \\
 & 0.5 * (1 + \gamma (\text{score} + 6)) + 0.5 * (1 + \gamma (\text{score} + 6)) \} \\
 = 6.4 & \quad (\text{for } \gamma = 0.9)
 \end{aligned}$$

Mini-Tetris: Bellman Backup

$$\theta^{(0)} = (-1, -1, -1, -1, -2, -2, -2, -3, -2, 20)$$

Bellman back-ups for the second state in S' :

$$\begin{aligned}
 V(\text{state}) = \max \{ & 0.5 * (1 + \gamma V(\text{sink-state, V=0})) + 0.5 * (1 + \gamma V(\text{sink-state, V=0})), \\
 & 0.5 * (1 + \gamma V(\text{sink-state, V=0})) + 0.5 * (1 + \gamma V(\text{sink-state, V=0})), \\
 & 0.5 * (1 + \gamma V(\text{sink-state, V=0})) + 0.5 * (1 + \gamma V(\text{sink-state, V=0})), \\
 & 0.5 * (1 + \gamma \theta^\top \phi(\text{state})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{state})) \} \\
 & \quad \quad \quad (0,0,0,0, 0,0,0, 0, 0, 1) \quad \quad \quad (0,0,0,0, 0,0,0, 0, 0, 1) \\
 & \quad \quad \quad \rightarrow V = 20 \quad \quad \quad \rightarrow V = 20
 \end{aligned}$$

= 19

Mini-Tetris: Bellman Backup

$$\theta^{(0)} = (-1, -1, -1, -1, -2, -2, -2, -3, -2, 20)$$

Bellman back-ups for the third state in S' :

$$\begin{aligned}
 V(\text{grid with 2 yellow blocks}) &= \max \{ 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks, 1 blue block})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks})) \}, \\
 &\quad \begin{array}{c} (4,4,0,0, 0,4,0, 4, 0, 1) \\ \rightarrow V = -8 \end{array} \qquad \begin{array}{c} (4,4,0,0, 0,4,0, 4, 0, 1) \\ \rightarrow V = -8 \end{array} \\
 &\quad 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks, 1 blue block})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks})), \\
 &\quad \begin{array}{c} (2,4,4,0, 2,0,4, 4, 0, 1) \\ \rightarrow V = -14 \end{array} \qquad \begin{array}{c} (2,4,4,0, 2,0,4, 4, 0, 1) \\ \rightarrow V = -14 \end{array} \\
 &\quad 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks, 1 blue block})) + 0.5 * (1 + \gamma \theta^\top \phi(\text{grid with 3 yellow blocks})) \} \\
 &\quad \begin{array}{c} (0,0,0,0, 0,0,0, 0, 0, 1) \\ \rightarrow V = 20 \end{array} \qquad \begin{array}{c} (0,0,0,0, 0,0,0, 0, 0, 1) \\ \rightarrow V = 20 \end{array}
 \end{aligned}$$

$$= 19$$

Mini-Tetris: Bellman Backup

$$\theta^{(0)} = (-1, -1, -1, -1, -2, -2, -2, -3, -2, 20)$$

Bellman back-ups for the fourth state in S' :

$$\begin{aligned}
 V(\text{State}) = \max \{ & 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_1)) + 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_2)), \\
 & 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_3)) + 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_4)), \\
 & 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_5)) + 0.5 * (1 + \gamma \theta^\top \phi(\text{State}_6)) \}
 \end{aligned}$$

(6,6,4,0, 0,2,4, 6, 4, 1) $\rightarrow V = -34$ (6,6,4,0, 0,2,4, 6, 4, 1) $\rightarrow V = -34$
 (4,6,6,0, 2,0,6, 6, 4, 1) $\rightarrow V = -38$ (4,6,6,0, 2,0,6, 6, 4, 1) $\rightarrow V = -38$
 (4,0,6,6, 4,6,0, 6, 4, 1) $\rightarrow V = -42$ (4,0,6,6, 4,6,0, 6, 4, 1) $\rightarrow V = -42$

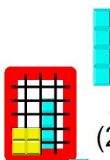
$$= -29.6$$

Mini-Tetris: Supervised Learning

After running the Bellman back-ups for all 4 states in S' we have:

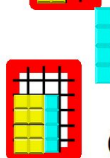
$$V(\text{state 1}) = 6.4$$

(2,2,4,0, 0,2,4, 4, 0, 1)



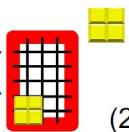
$$V(\text{state 2}) = 19$$

(4,4,4,0, 0,0,4, 4, 0, 1)



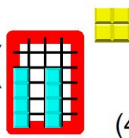
$$V(\text{state 3}) = 19$$

(2,2,0,0, 0,2,0, 2, 0, 1)



$$V(\text{state 4}) = -29.6$$

(4,0,4,0, 4,4,4, 4, 0, 1)



We now run supervised learning on these 4 examples to find a new θ :

$$\begin{aligned} \min_{\theta} & (6.4 - \theta^T \phi(\text{state 1}))^2 \\ & + (19 - \theta^T \phi(\text{state 2}))^2 \\ & + (19 - \theta^T \phi(\text{state 3}))^2 \\ & + ((-29.6) - \theta^T \phi(\text{state 4}))^2 \end{aligned}$$

Running least squares gives:

$$\theta^{(1)} = (0.195, 6.24, -2.11, 0, -6.05, 0.13, -2.11, 2.13, 0, 1.59)$$

Guarantees?

FA with Q-values