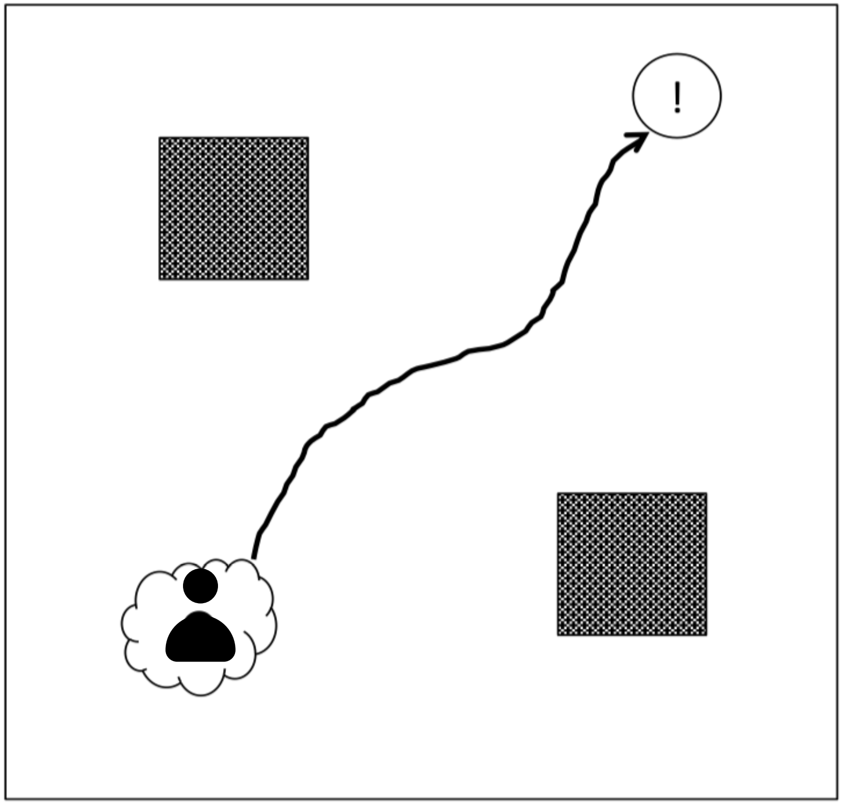
50.007
Machine Learning

Lu, Wei

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

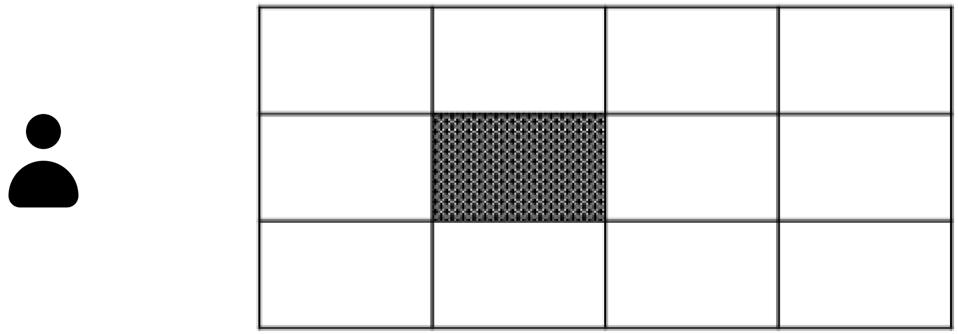
Reinforcement Learning (II)

Learn How to Act



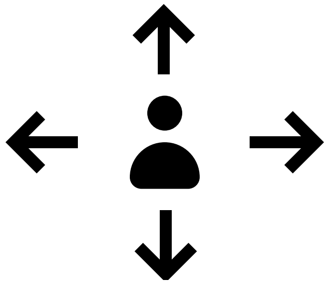
How do we teach a robot how to act optimally in a complex environment?

Block World
Environment

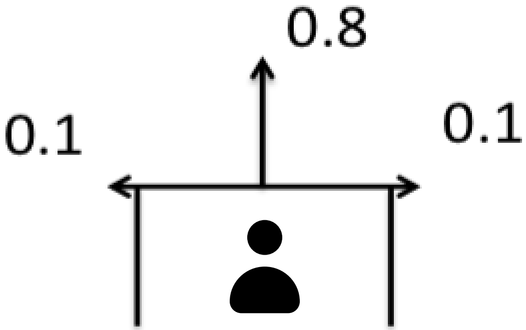


The robot can take an action from a set of predefined possible actions at each state.

$$\text{There is a set of actions } A$$



Block World
Transition Probabilities



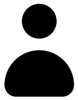
If the robot moves towards a particular direction, there is a 0.8 chance that it would reach the block in front, and there is a 0.1 chance to reach a state to its left (right).

A transition probability function

$$T(s, a, s') = p(s' | s, a)$$

Block World

Rewards



-0.6	+1.2	+0.1	+1.0
-0.1		-0.1	-1.0
+0.9	-0.7	-2.0	+1.3

Each block is associated with a reward. The two blocks at the upper-right corner are assigned rewards +1 and -1.

The reward is $R(s)$ for each state.

action

In general it can be defined as $R(s, a, s')$

old state

new state

Markov Decision Process

-0.6	+1.2	+0.1	+1.0
-0.1		-0.1	-1.0
+0.9	-0.7	-2.0	+1.3

- a set of states S

- a set of actions A

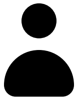
- a transition probability function $T(s, a, s') = p(s'|s, a)$

- a reward function $R(s, a, s')$ (or just $R(s')$)

How do we learn the policy?

Block World

Utility (Long Term Reward)



-0.6	+1.2	+0.1	+1.0
0.1		-0.1	-1.0
+0.9	-0.7	-2.0	+1.3

$$U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots = \sum_{t=0}^{\infty} \gamma^t R(s_t)$$

$$\underbrace{\frac{R_{min}}{1-\gamma}}_{\text{Lower Bound}} = \sum_{t=0}^{\infty} \gamma^t \underbrace{R_{min}}_{\text{Smallest Reward}} \leq U([s_0, s_1, s_2, \ldots]) \leq \sum_{t=0}^{\infty} \gamma^t \underbrace{R_{max}}_{\text{Largest Reward}} = \underbrace{\frac{R_{max}}{1-\gamma}}_{\text{Upper Bound}}$$

Policy, Value, Q-Value

$\pi^*(s)$

The *optimal policy* $\pi^*(s)$ specifies

the optimal action we should take in state s.

$V^*(s)$

The *value* of state s under the optimal policy π^*

$Q^*(s, a)$

The Q-*value* of state s

and action a under the optimal policy π^*

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s, a) = Q^*(s, \pi^*(s))$$

$$Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s, a) = Q^*(s, \pi^*(s))$$

$$Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

$$V^*(s) = \max_a \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

Value Iteration

1. Start with $V_0^*(s) = 0$, for all $s \in S$

2. Given V_i^*, calculate the values for all states $s \in S$

$$V_{i+1}^*(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_i^*(s')]$$

3. Repeat the above until convergence

There is a guarantee that this process will converge

Learning Optimal Policy

Step 1
Run Value Iteration Algorithm

Step 2
Calculate the Q values

$$Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

Step 3
Find the optimal action for each state

$$\pi^*(s) = \arg\max_a Q^*(s, a)$$

Value Iteration

Step 1

Since the procedure relies on Q-values, is it possible to design an algorithm that directly computes these Q-values?

Step 2
Find the optimal action for each state

$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

Value Iteration

Step 1

Since the procedure relies on Q-values, is it possible to design an algorithm that directly computes these Q-values?

Step 2
Find the optimal action for each state

$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s,a) = Q^*(s,\pi^*(s))$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s,a) = Q^*(s, \pi^*(s))$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

Can we have an equation for which both sides involve the Q terms only?

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s,a) = Q^*(s, \pi^*(s))$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

Can we have an equation for which both
sides involve the Q terms only?

Policy, Value, Q-Value

$$V^*(s) = \max_a Q^*(s,a) = Q^*(s, \pi^*(s))$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

$$V^*(s') = \boxed{\max_{a'} Q^*(s',a')}$$

Policy, Value, Q-Value

$$Q^*(s,a) \leftarrow \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')]$$

Policy, Value, Q-Value

1. Start with $Q_0^*(s,a) = 0$ for all $s \in S, a \in A$

2. Given $Q_i^*(s,a)$, calculate the Q-values for all states (depth $i+1$) and for all actions a:

$$Q_{i+1}^*(s,a) \leftarrow \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q_i^*(s',a')]$$

3. Repeat the above step until convergence.

Q-Value Iteration

1. Start with $Q_0^*(s, a) = 0$ for all $s \in S, a \in A$

2. Given $Q_i^*(s, a)$, calculate the Q-values for all states (depth $i + 1$) and for all actions a:

$$Q_{i+1}^*(s, a) \leftarrow \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma \max_{a'} Q_i^*(s', a')]$$

3. Repeat the above step until convergence.

Q-Value Iteration

1. Start with $Q_0^*(s,a) = 0$ for all $s \in S, a \in A$

2. Given $Q_i^*(s,a)$, calculate the Q-values for all states (depth $i+1$) and for all actions a:

$$Q_{i+1}^*(s,a) \leftarrow \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q_i^*(s',a')]$$

3. Repeat the above step until convergence.

Again, there is a guarantee it will converge.

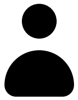
Learning Optimal Policy

Step 1
Run Q-Value Iteration Algorithm

Step 2
Find the optimal action for each state

$$\pi^*(s) = \arg\max_a Q^*(s, a)$$

Markov Decision Process

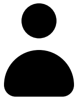


-0.6	+1.2	+0.1	+1.0
-0.1		-0.1	-1.0
+0.9	-0.7	-2.0	+1.3

- a set of states S
- a set of actions A
- a transition probability function $T(s, a, s') = p(s'|s, a)$
- a reward function $R(s, a, s')$ (or just $R(s')$)

How do we learn the policy?

Reinforcement Learning



-0.6	+1.2	+0.1	+1.0
-0.1		-0.1	-1.0
+0.9	-0.7	-2.0	+1.3

- a set of states S

- a set of actions A

- a transition probability function $T(s, a, s') = p(s'|s, a)$

- a reward function $R(s, a, s')$ (or just $R(s')$)

How do we learn the policy?

Reinforcement Learning

How do we deal with the unknown transition probabilities and the reward functions?

- a set of states S
- a set of actions A
- a transition probability function $T(s, a, s') = p(s'|s, a)$
- a reward function $R(s, a, s')$ (or just $R(s')$)

How do we learn the policy?

Reinforcement Learning

How do we deal with the unknown transition probabilities and the reward functions?

Imagine the robot is moving around...

$$T(s'|s,a) = \frac{\mathrm{Count}(s,a,s')}{\mathrm{Count}(s,a)} \qquad R(s,a,s') = \frac{\sum_t R_t(s,a,s')}{\mathrm{Count}(s,a,s')}$$

Model-based Approach

Estimating the model parameters first

Reinforcement Learning

How do we deal with the unknown transition probabilities and the reward functions?

Imagine the robot is moving around...

$$T(s'|s,a) = \frac{\text{Count}(s,a,s')}{\text{Count}(s,a)} \qquad R(s,a,s') = \frac{\sum_t R_t(s,a,s')}{\text{Count}(s,a,s')}$$

Model-free Approach

Does not explicitly estimate parameters

Model-Free Approach

A Simple Game

Flipping a coin, if it is head (H), you receive $200; otherwise you pay $100. Would you like to play the game?

Model-Free Approach

A Simple Game

Flipping a coin, if it is head (H), you receive $200, otherwise you pay $100. Would you like to play the game?

$$p(\mathrm{T}) \times (-100) + p(\mathrm{H}) \times (+200)$$

Model Parameters

Model-Free Approach

H

H

T

T

H

T

T

T

T

T

T

$$p(\mathrm{H}) = 0.3$$

$$p(\mathrm{T}) = 0.7$$

$$p(\mathrm{T}) \times (-100) + p(\mathrm{H}) \times (+200) =$$

$$= -70 + 60 = \text{- } 10$$

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	+100	+100

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0
	T	-200	
	H	$+100$	
	T	-200	
	T	-200	$\vdots$
	T	-200	
	T	-200	
9	T	-200	
10	T	-200	-10

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0
	$\vdots$	$\vdots$	$\vdots$
$k-1$	T	-200	v

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0
	$\vdots$	$\vdots$	$\vdots$
$k-1$	T	-200	v

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0
	$\vdots$	$\vdots$	$\vdots$
$k-1$	T	-200	v
k	H	$+100$	??

Model-Free Approach

	Outcome	Gain/Loss	Average Return
1	H	$+100$	$+100$
2	H	$+100$	$+100$
3	T	-200	0
	$\vdots$	$\vdots$	$\vdots$
$k-1$	T	-200	v
k	H	$+100$	$\frac{v\times(k-1)+(+100)}{k}$

Model-Free Approach

$$v_{new} \leftarrow \frac{v_{old} \times (k-1) + (+100)}{k}$$

Model-Free Approach

$$v_{new} \leftarrow v_{old} + \frac{1}{k}((+100) - v_{old})$$

Model-Free Approach

$$v_{new} \leftarrow v_{old} + \frac{1}{k}((+100) - v_{old})$$

Current estimate of Q-Value

From a newly collected sample

Model-Free Approach

"Learning Rate"

$$v_{new} \leftarrow v_{old} + \frac{1}{k}\boxed{((+100) - v_{old})}$$

"Gradient"

Updating Q-Values

We have arrived at the following model-free based estimation of the Q-values (after taking action a from state s)

$$Q(s,a)_{new} \leftarrow Q(s,a)_{old} + \frac{1}{k}\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)_{old}\right]$$

Q Learning

Collect a sample: s, a, s' and $R(s, a, s')$.

Update Q-values, by incorporating the new sample into a running average over samples:

$$Q(s,a)_{new} \leftarrow Q(s,a)_{old} + \frac{1}{k}\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)_{old}\right]$$

Q Learning

Collect a sample: s, a, s' and $R(s, a, s')$.

⬇

Update Q-values, by incorporating the new sample into a running average over samples:

$$Q(s,a)_{new} \leftarrow Q(s,a)_{old} +$$

$$\frac{1}{k}\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)_{old}\right]$$

$\uparrow$

Can be replaced by α

Q Learning

Collect a sample: s, a, s' and $R(s, a, s')$.

How shall we choose the action?

Update Q-values, by incorporating the new sample into a running average over samples:

$$Q(s,a)_{new} \leftarrow Q(s,a)_{old} + \frac{1}{k}\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)_{old}\right]$$

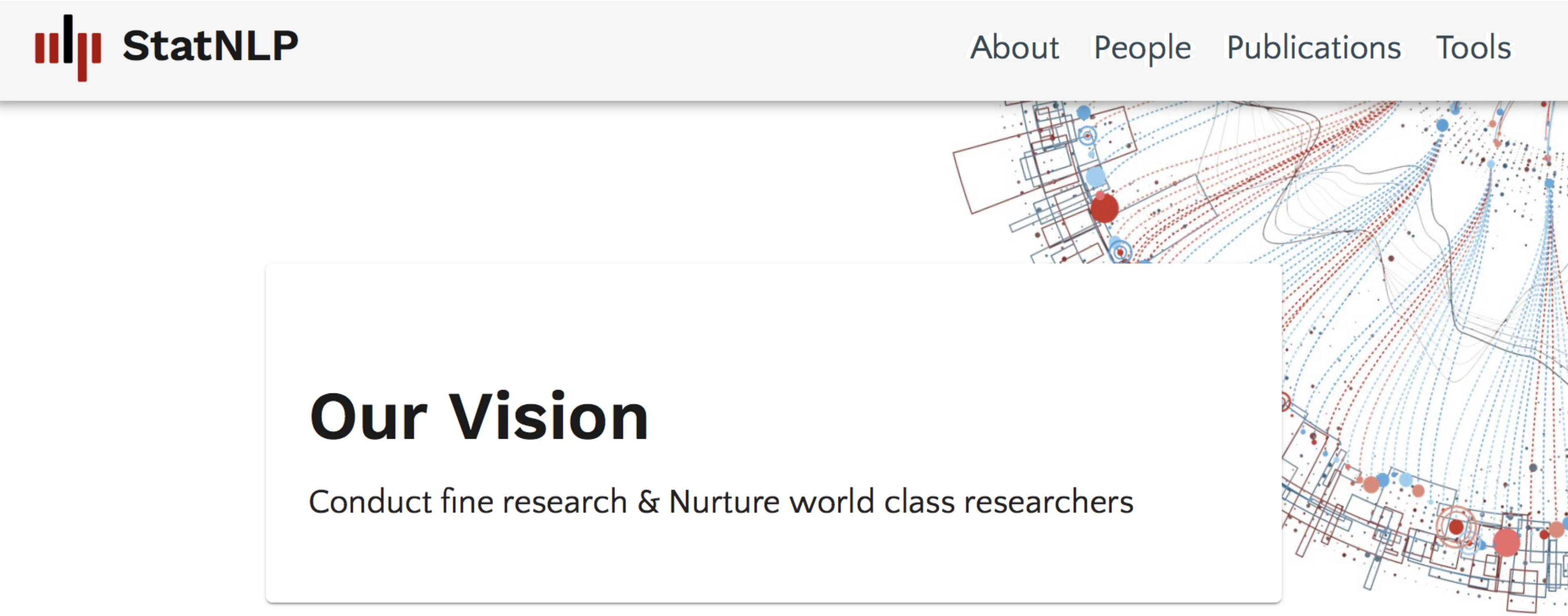
Exploration vs Exploitation

Collect a sample: s, a, s' and $R(s, a, s')$.

Initially, we may randomly pick the action a (exploration). Later, when we are more confident about the learned Q-values, we may follow the action based on Q-values (exploitation)

$$Q(s,a)_{new} \leftarrow Q(s,a)_{old} +$$
$$\frac{1}{k}\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)_{old}\right]$$

See you ...



Hope to see many of you at NLP next Summer!