ICS 691D: REINFORCEMENT LEARNING (RL)

Peter Y. Washington, PhD

Assistant Professor, ICS

August 31, 2022

UPCOMING ASSIGNMENTS DUE

- Wednesday September 7: Haber reflection
- Monday September 12: Akalin reflection
- Wednesday September 14: Kohli reflection

LITERATURE REVIEW AND DISCUSSION TOPICS RELEASED

- See comments section of your returned assignment on Laulima
  - If required, send me an email clarifying your topic

- You may switch with a student, but email me if you both agree to do so

- Earlier discussions will be graded much more leniently

IN CASE YOU WANT TO SWITCH TOPICS

5 minutes of class time to switch topics with a classmate for any reason

Just email me the switch so I can make it official

GUIDELINES FOR DISCUSSION LEADING

- Lead class discussion for 1 class about the general discussion topic for that day as well as the specific reading for that day's class
- Come up with discussion questions beforehand and ask the class (10 recommended; can be any balance of general vs. reading-specific questions)
- Questions should include at least one technical, design, and ethical question
- Call on classmates to answer, keep the conversation going, and keep it interesting
- Discussion does not need to exclusively consist of calling on people to answer: you may employ novel discussion leading topics (we will demonstrate an example in class today), e.g., lining people up on a spectrum of agreeing / disagreeing on a topic, breaking people into groups and having them present, creating a debate, or any other creative medium you wish
- Will require a thorough reading of the paper
- The combined presentation + leading of discussion section should be ~60 minutes
- Can include up to 15 minutes of video integrated into your presentation (we will demonstrate an example in class today)
- **Submit your presentation slides, which should include your discussion questions at the end, on Laulima by 3pm the day before your presentation so I can prepare complementary discussion / lecture which doesn't overlap with what you present**

GOOD EXAMPLE OF REFLECTION TOPICS (AND THEREFORE DISCUSSION QUESTION TOPICS)

- What you agreed/disagreed with and why
- Critique of any section, such as (1) the claims in the Introduction and Discussion, (2) the methods used, (3) the evaluation methods used in the Results, etc.
- How you might repurpose something you learned for an area that is of interest to you (including your research proposal assignment)
- How the paper connects to other readings in the class
- The ethical implications of the described technology (would be good to connect to one of the formal ethical frameworks we discussed)
- How the described technology could be designed for real world use and factors to consider when doing so
- Discussions of real-world implementations of the technique (such as companies in industry)
- …

CONCRETE EXAMPLES OF GOOD DISCUSSION QUESTIONS

All the discussion question examples provided in the first 3 lectures

Sample Topic: Chatbots for Mental Healthcare (not a topic in this class)

1. Technical/Design: What are different AI approaches which could be used to develop a mental health chatbot?
2. Design: What training data should be used? What are potential biases that these data could incur?
3. Ethics: What can go wrong with an AI chatbot?
4. Policy/Ethics: What regulations should governments impose on mental health chatbots?
5. Ethics: What are possible implications of the commercialization of chatbots? How would companies make money with such a system, and how could that impact the end user?
6. Technical/Design: How might a chatbot system be evaluated? Which quantitative metrics should be used?
7. Technical: What are some methodological limitations and/or areas of future work to improve the chatbot system described in today's paper?
8. Technical: How might rule-based systems, such as that described in the paper, be combined with statistical learning approaches to create a system that benefits from the "best of both worlds"?

…

OUTLINE

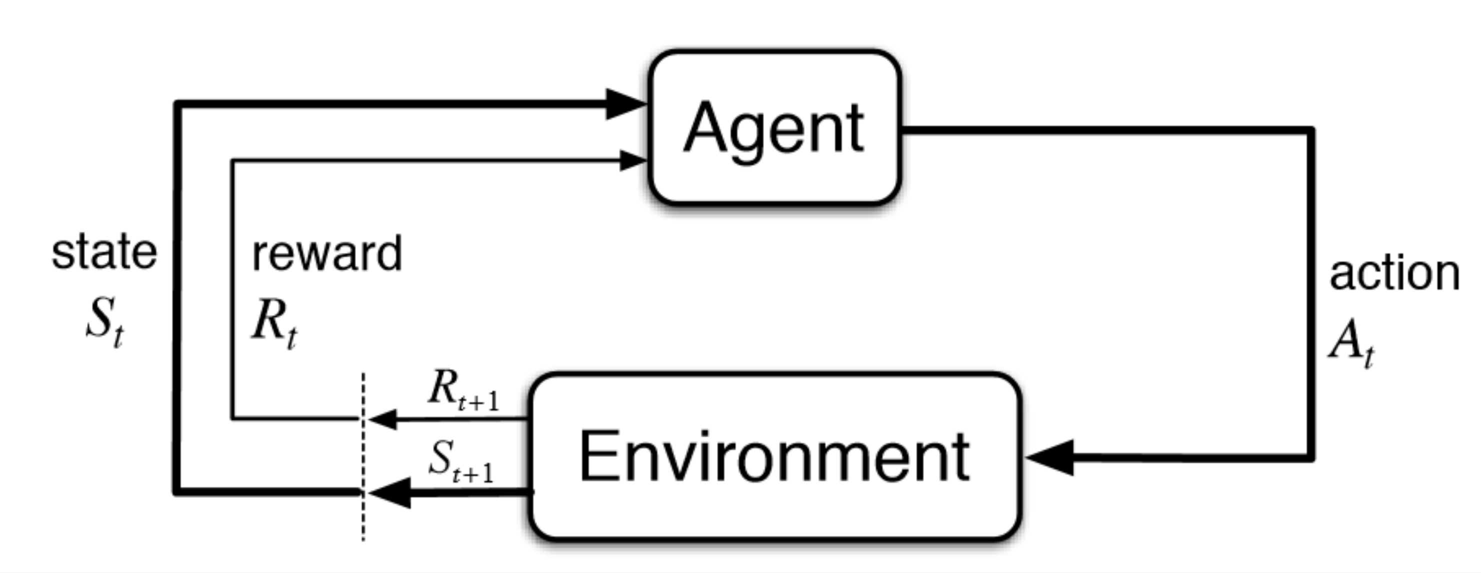
- RL overview
- RL applications
- Deep RL overview
- Deep RL applications
- Discussion

OUTLINE

- RL overview
- RL applications
- Deep RL overview
- Deep RL applications
- Discussion

REINFORCEMENT LEARNING (RL)

Learning by experience (RL) vs learning by examples (supervised learning)



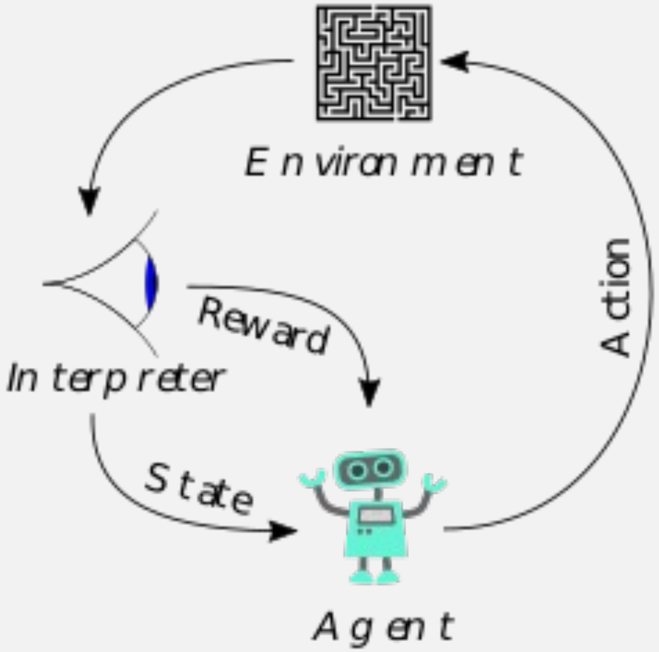
https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

WHEN IS RL USED?

- Autonomous vehicles (cars, spacecraft, aircraft, satellites, submarines, …)
- Robotics
- Recommender systems
- AI for gaming
  - Usually as a "test bed" for more real-world ML
- Decision support (e.g., finance/trading)
- Artificial General Intelligence (AGI)
  - Maybe one day

COMPONENTS OF AN RL SYSTEM

- **Agent**: the RL actor (robot, car, etc)
- **States**: all possible configurations of the agent
- **Actions**: the possible actions of the agent
- **Reward**: feedback from the environment
- **Policy**: function returning new state given the current state and an action
- **Value**: function returning quantification of reward in the future given the current state and an action



Wikipedia

TYPES OF ENVIRONMENTS

- Fully Observable (Chess) vs. Partially Observable (Poker)
- Single Agent (Atari) vs. Multi Agent (Self-Driving Cars)
- Deterministic (Chess) vs. Stochastic (Self-Driving Cars)
- Discrete (Chess) vs. Continuous (Robotic Navigation)

ULTIMATE GOAL: MAXIMIZE REWARD

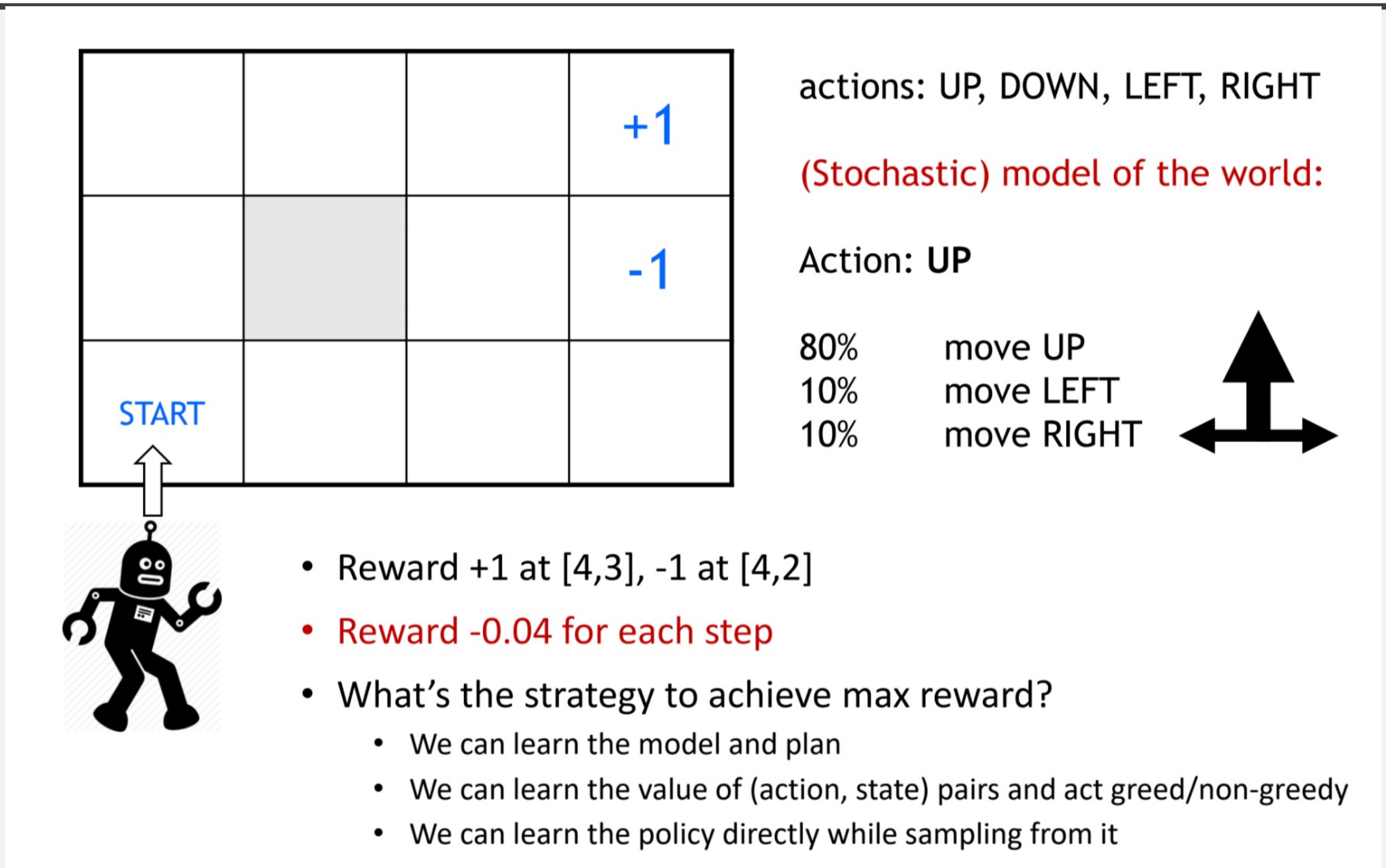
Future Reward:

$$R_t = r_t + r_{t+1} + r_{t+2} + \cdots + r_n$$

Discounted Future Reward:

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{n-t} r_n$$

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



actions: UP, DOWN, LEFT, RIGHT

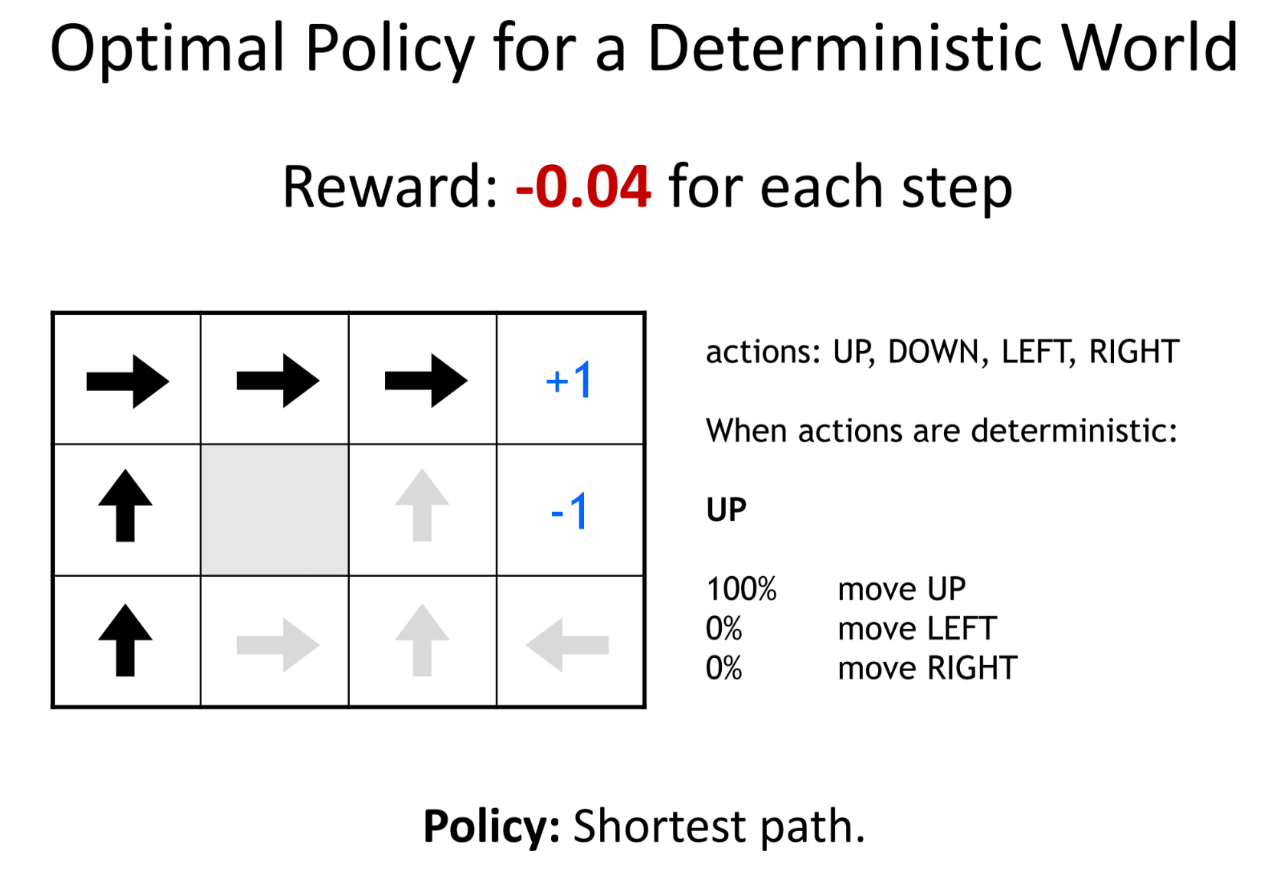
(Stochastic) model of the world:

Action: **UP**

80%	move UP
10%	move LEFT
10%	move RIGHT

- Reward +1 at [4,3], -1 at [4,2]
- Reward -0.04 for each step
- What's the strategy to achieve max reward?
  - We can learn the model and plan
  - We can learn the value of (action, state) pairs and act greed/non-greedy
  - We can learn the policy directly while sampling from it

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



Optimal Policy for a Deterministic World

Reward: **-0.04** for each step

actions: UP, DOWN, LEFT, RIGHT

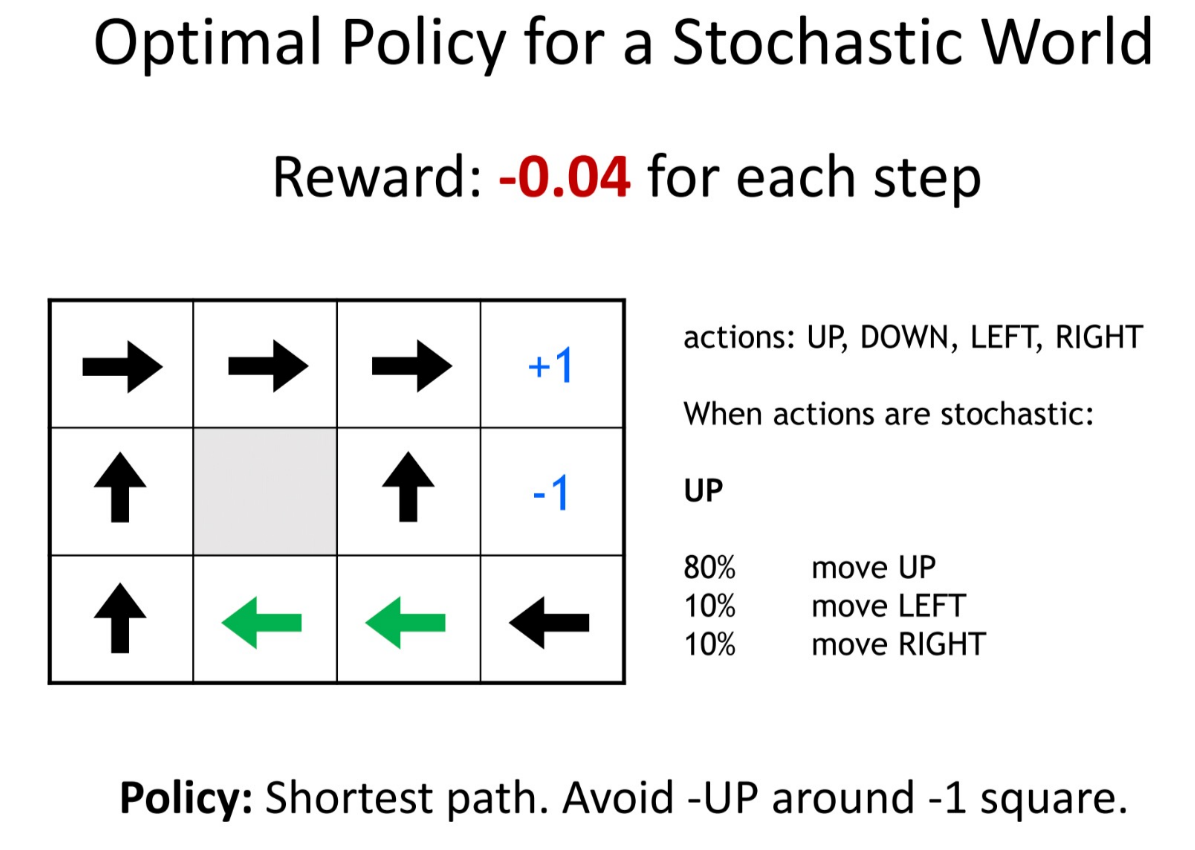
When actions are deterministic:

UP

100%	move UP
0%	move LEFT
0%	move RIGHT

Policy: Shortest path.

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



Optimal Policy for a Stochastic World

Reward: **-0.04** for each step

→	→	→	+1
↑	▨	↑	-1
↑	←	←	←

actions: UP, DOWN, LEFT, RIGHT

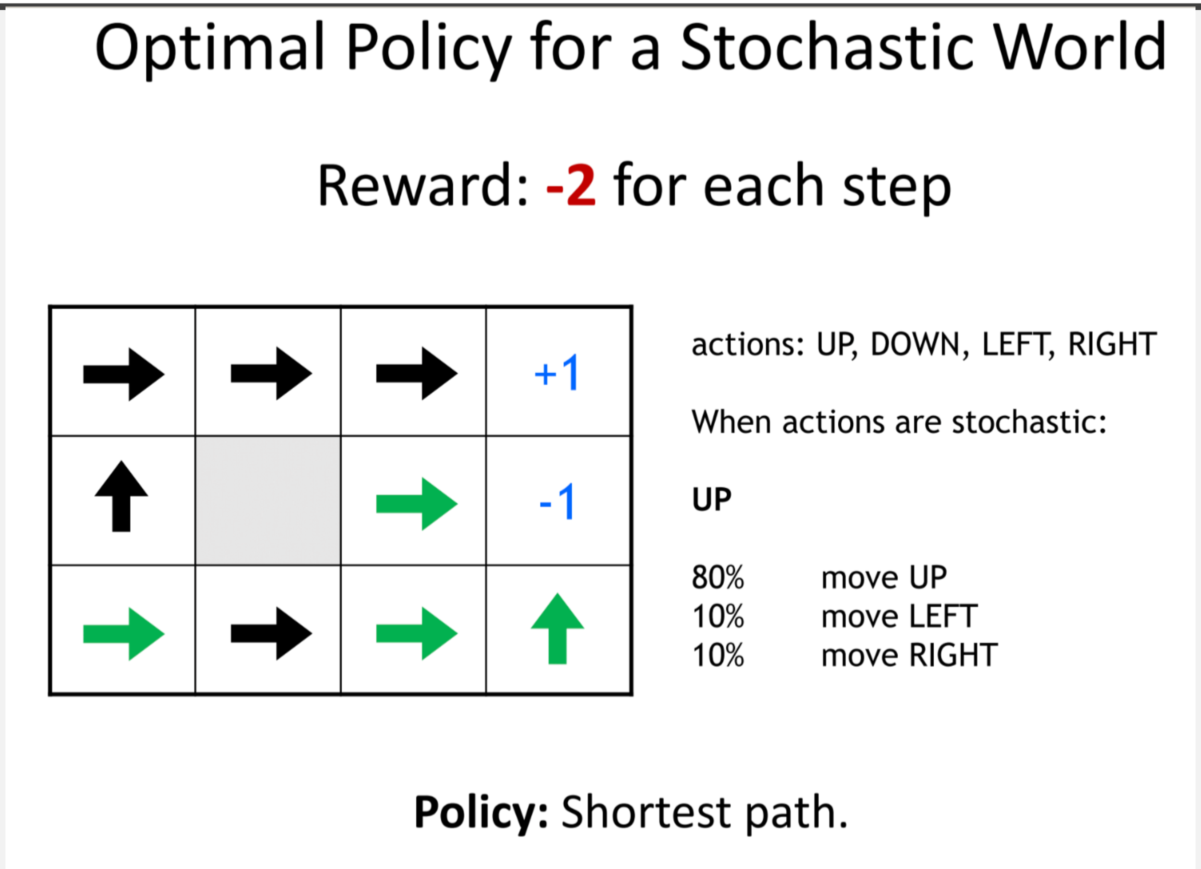
When actions are stochastic:

UP

80%	move UP
10%	move LEFT
10%	move RIGHT

Policy: Shortest path. Avoid -UP around -1 square.

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



Optimal Policy for a Stochastic World

Reward: **-2** for each step

actions: UP, DOWN, LEFT, RIGHT

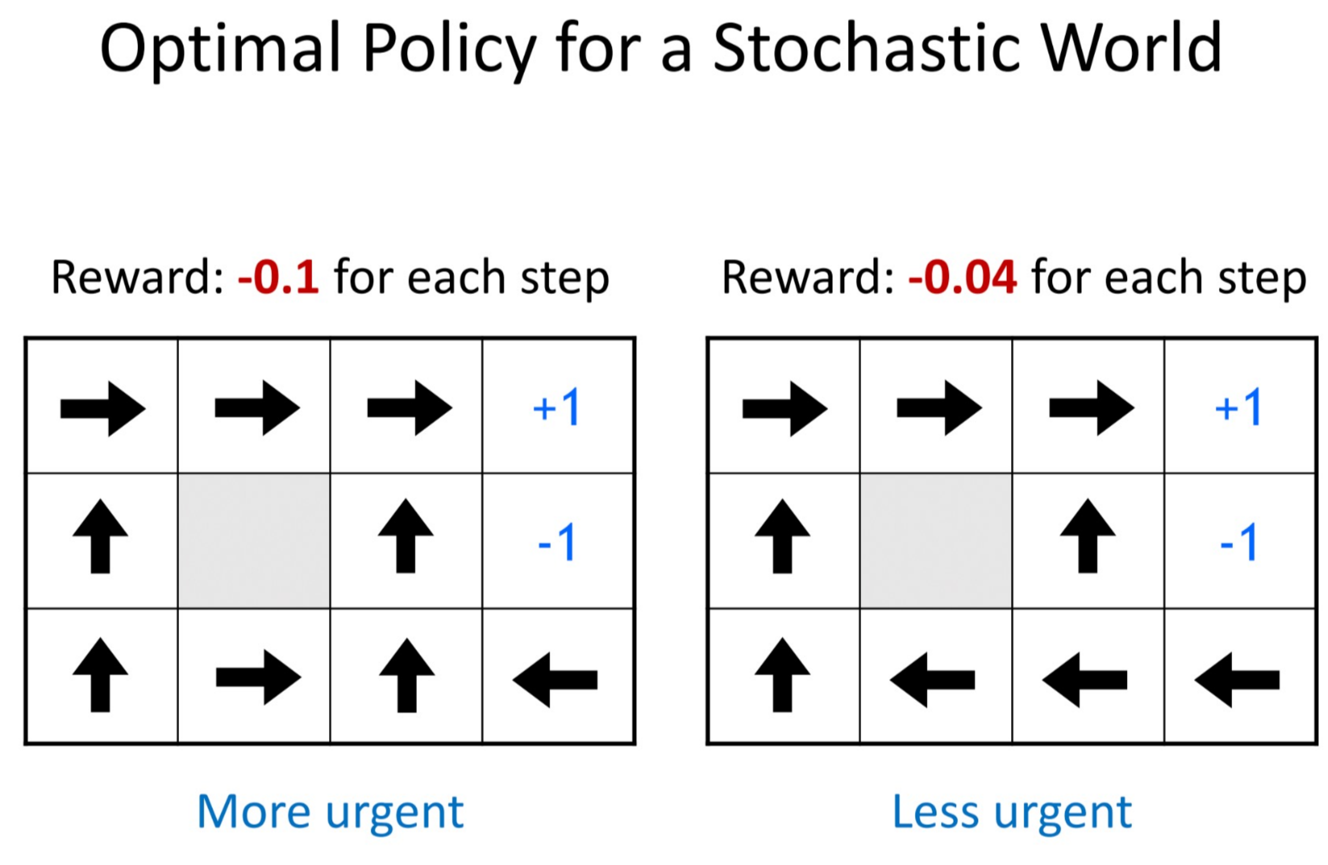
When actions are stochastic:

UP

80%	move UP
10%	move LEFT
10%	move RIGHT

Policy: Shortest path.

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



Optimal Policy for a Stochastic World

Reward: **-0.1** for each step

→	→	→	+1
↑	▨	↑	-1
↑	→	↑	←

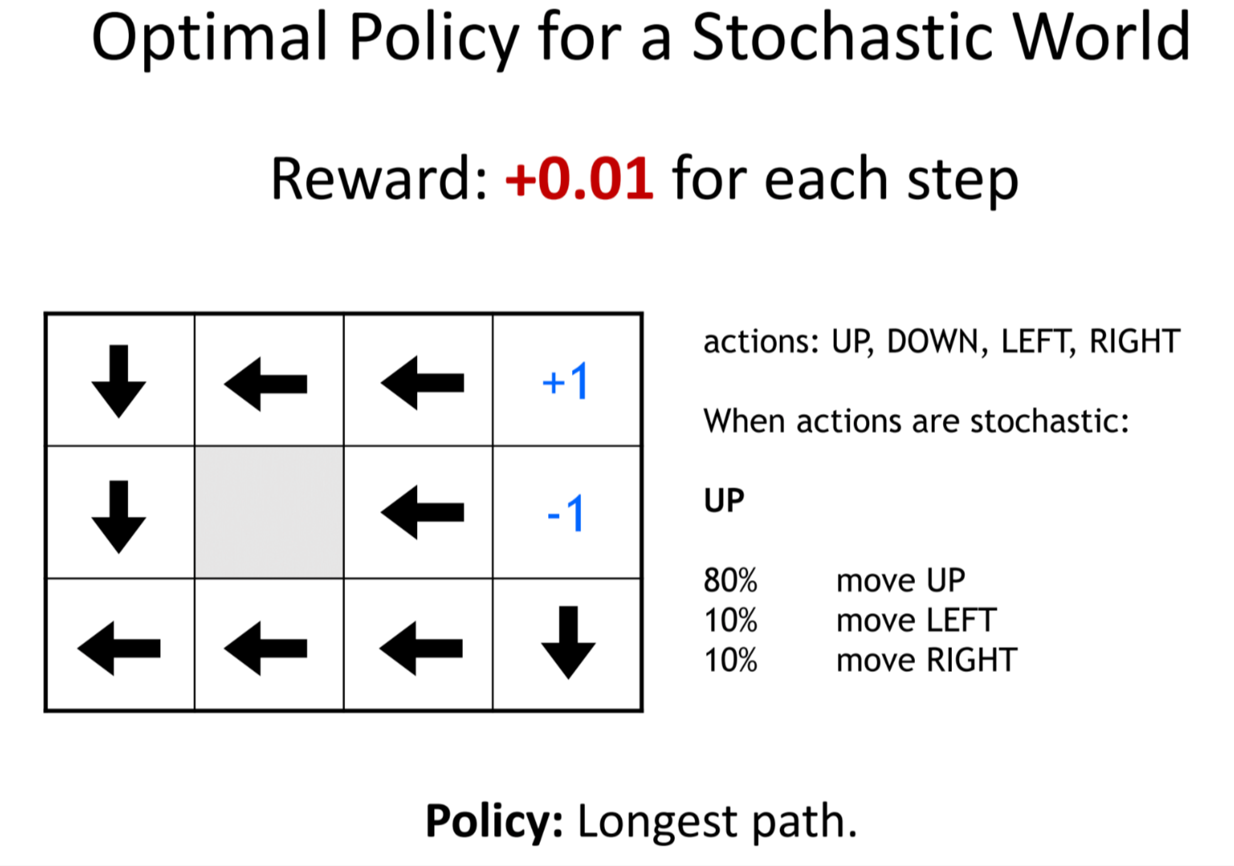
More urgent

Reward: **-0.04** for each step

→	→	→	+1
↑	▨	↑	-1
↑	←	←	←

Less urgent

REWARDS ARE DEFINED BY HUMANS: DEMONSTRATIVE EXAMPLE



Optimal Policy for a Stochastic World

Reward: **+0.01** for each step

↓	←	←	+1
↓	▨	←	-1
←	←	←	↓

actions: UP, DOWN, LEFT, RIGHT

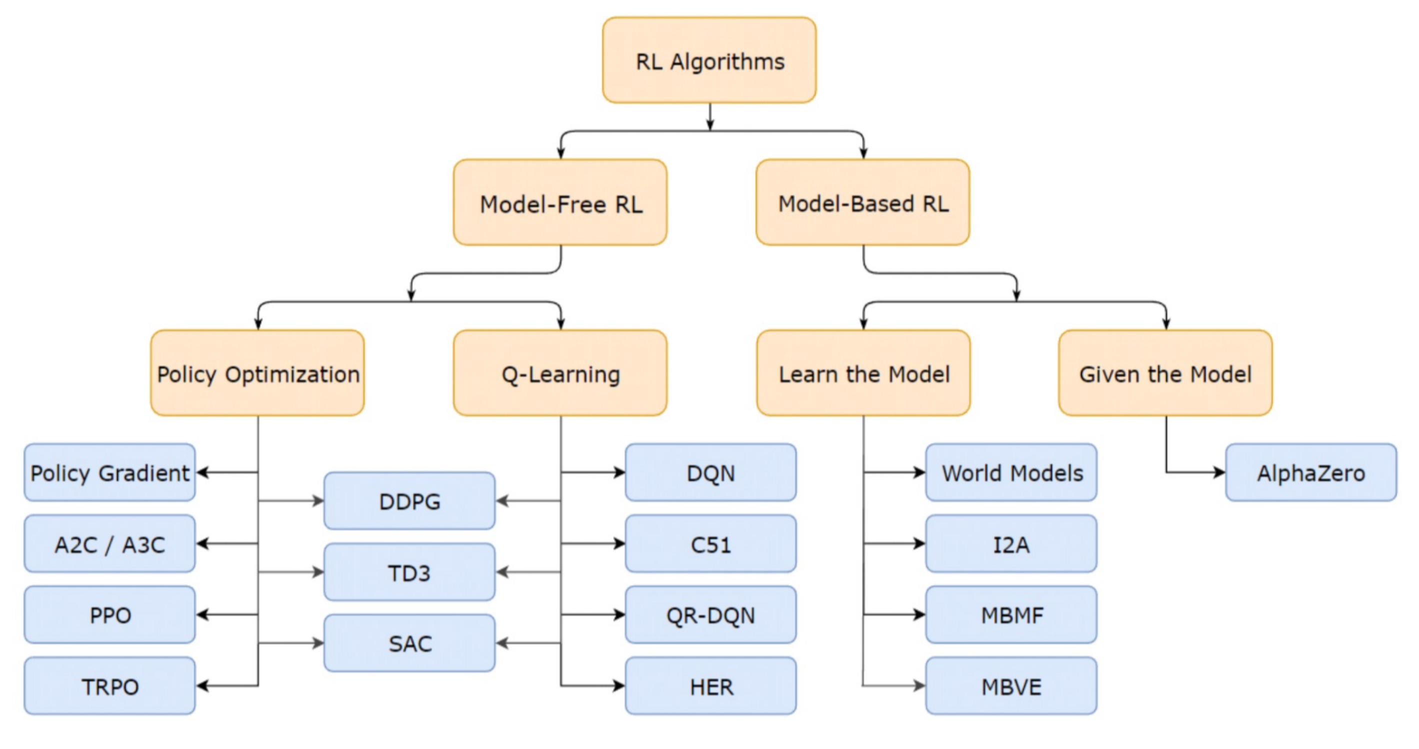
When actions are stochastic:

UP

80%	move UP
10%	move LEFT
10%	move RIGHT

Policy: Longest path.

MANY TYPES OF RL METHODS



https://spinningup.openai.com/

MODEL-BASED RL: MARKOV DECISION PROCESS (MDP)

(S, A, R, P)

S: states

A: actions

R: reward function, R(s, s', a)

P: transition function, $P(s, s', a) = P(s_{t+1}=s' \mid s_t = s, a_t = a)$

POLICIES AND VALUES

Policy (usually denoted as π): $\pi(a|s)$ is the probability of taking action a when currently in state s

Value function: $V_{\pi}(s)$ is the expected total return when starting in state s and following policy π thereafter

BELLMAN EQUATION

$$V(x) = \max_{a \in \Gamma(x)} \{F(x,a) + \beta V(T(x,a))\}$$

$$\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \,\middle|\, S_t = s\right] \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \left[r + \gamma \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \,\middle|\, S_{t+1} = s'\right]\right] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \Big[r + \gamma v_\pi(s')\Big], \qquad (3.12)
\end{aligned}$$

Intuitively:

- In general, the "value" of a decision problem at a certain point in time in terms of the payoff from some initial choices and the "value" of the remaining decision problem that results from those initial choices
- The value of the start state is the discounted value of the expected next state plus the reward expected along the way
- Averages over all possible next states, weighted by their probability of occurring

Reinforcement Learning: An Introduction, Richard Sutton and Andrew Barto

BRUTE FORCE SOLUTION

- For each possible policy, sample returns while following it
- Choose the policy with the largest expected return

BRUTE FORCE SOLUTION

- For each possible policy, sample returns while following it
- Choose the policy with the largest expected return

Issues:

- The number of policies can be large or infinite
- Variance of returns can be large, requiring large number of samples to estimate the return of any given policy

MODEL-BASED VS MODEL-FREE RL

- Model-based: explicitly learn the transition function P and the reward function R

- Model-free: learn a policy directly without necessarily understanding the world via P and R

MODEL-BASED RL: POLICY ITERATION

Start with an arbitrary policy **π**, then iteratively update the policy.

Repeat until convergence:

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s') \left(R_{\pi(s)}(s, s') + \gamma V(s') \right)$$

$$\pi(s) := \mathrm{argmax}_a \left\{ \sum_{s'} P_a(s, s') \left(R_a(s, s') + \gamma V(s') \right) \right\}$$

Act by sampling the policy

MODEL-BASED RL: VALUE ITERATION

Iteratively update the state-value function.

Repeat until convergence:

$$V_{i+1}(s) := \max_a \left\{ \sum_{s'} P_a(s, s') \left(R_a(s, s') + \gamma V_i(s') \right) \right\}$$

Act by choosing the best action in a state

MODEL-FREE RL: Q-LEARNING

- Q(s, a) is the expected return starting at state s, taking action a, then thereafter following policy π

- Best possible future return when performing action a in state s

- Q function is the Action-Value function for policy π

- In general: model-free RL involves predicting the value function of a certain policy without having a concrete model of the environment

Q-LEARNING

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

Compute Q function for all state-action pairs

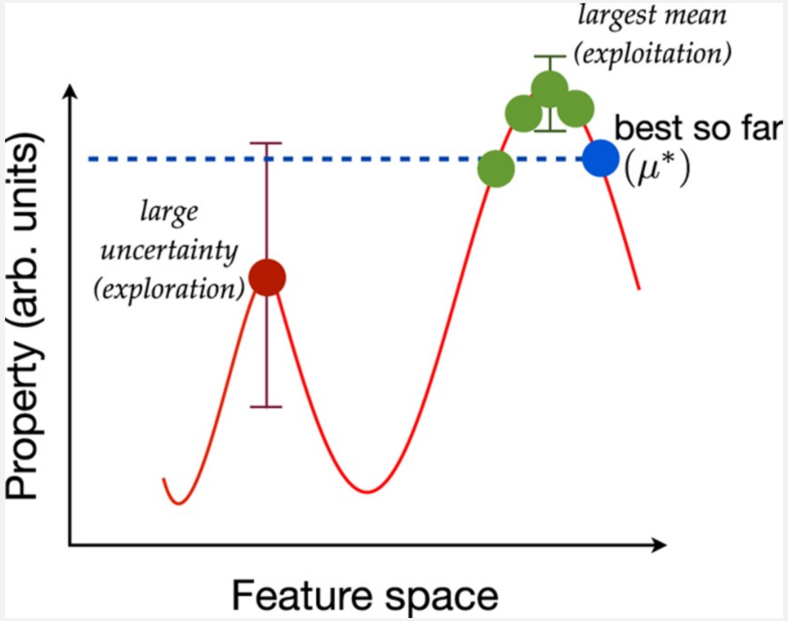
After Q function is learned, act by selecting a for current state which has the highest Q value

Q-LEARNING

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Bigg)}^{\text{temporal difference}}$$

FUNDAMENTAL TRADEOFF: EXPLORATION VS. EXPLOITATION

- Exploration: sample the global search space; helps avoid local optima
- Exploitation: maximize with a promising local region of the search space to fully optimize the solution



Balachandran et al. *Scientific Reports*. 2016.

OUTLINE

- RL overview
- RL applications
- Deep RL overview
- Deep RL applications
- Discussion

SELF-DRIVING CARS

Agent:

States:

Actions:

Rewards:

SELF-DRIVING CARS

Agent: the car

States: (Crash or No Crash, Lawful or Unlawful, distance from destination)

Actions: acceleration, deceleration, steering wheel angle change

Rewards: -10000 for Crash, +10 for No Crash, -100 for Unlawful, …

TARGETED ADVERTISEMENT RECOMMENDATION

Agent:

States:

Actions:

Rewards:

TARGETED ADVERTISEMENT RECOMMENDATION

Agent: the recommender system

States:

Actions:

Rewards:

OUTLINE

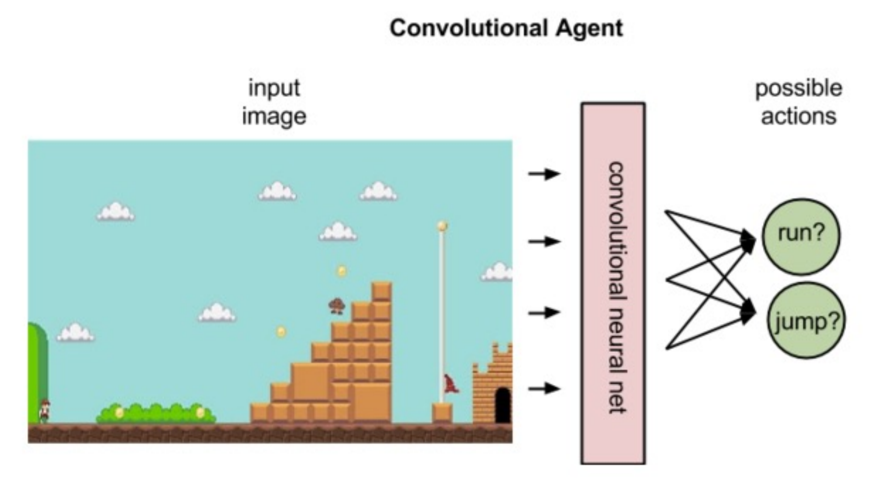
- RL overview
- RL applications
- Deep RL overview
- Deep RL applications
- Discussion

DEEP RL

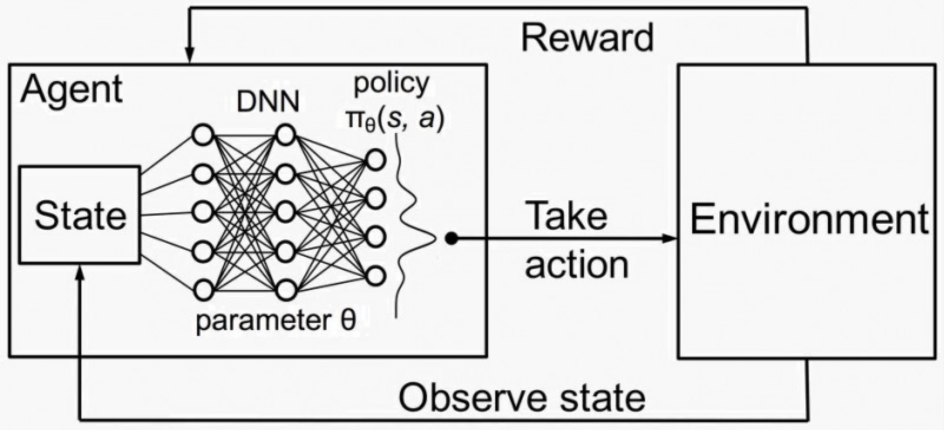
Replace parts of the RL process with DL

DEEP RL

Representation matters!

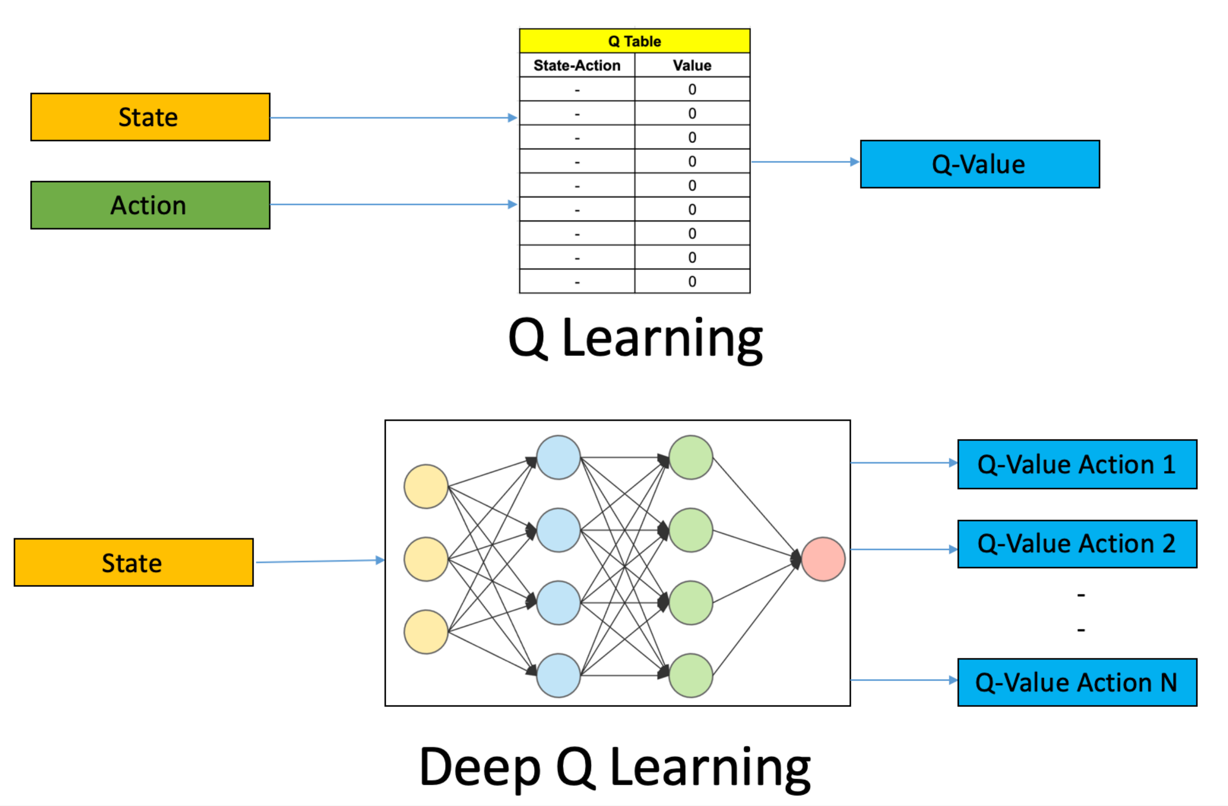


https://wiki.pathmind.com/deep-reinforcement-learning



https://medium.com/@vishnuvijayanpv/deep-reinforcement-learning-value-functions-dqn-actor-critic-method-backpropagation-through-83a277d8c38d

DEEP Q-LEARNING



Q Table	
State-Action	Value
-	0
-	0
-	0
-	0
-	0
-	0
-	0
-	0
-	0

$$Q^{new}(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t,a_t)}_{\text{old value}}\Bigg)}^{\text{temporal difference}}$$

https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

“WORLD MODELS”

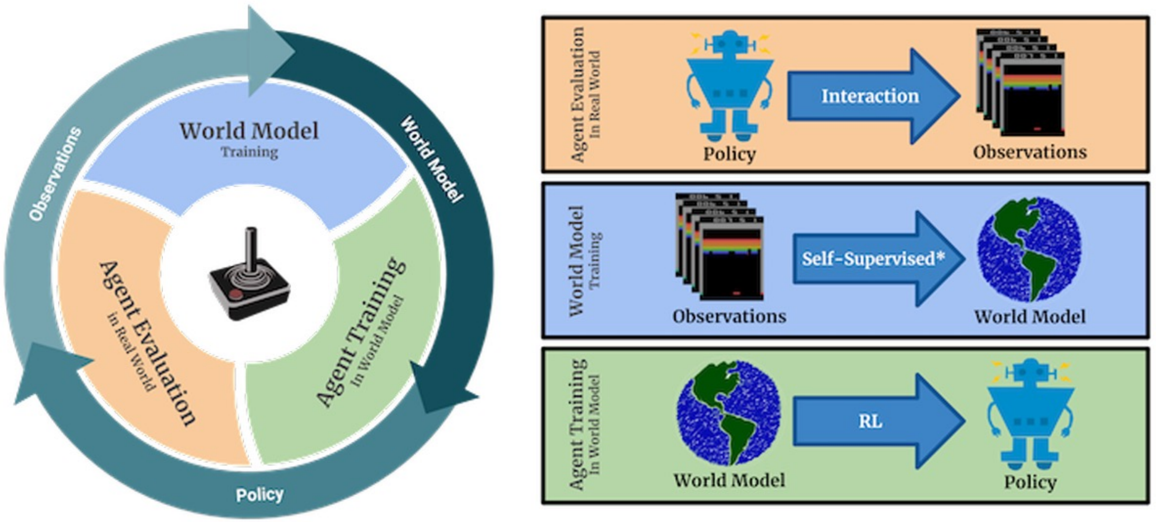
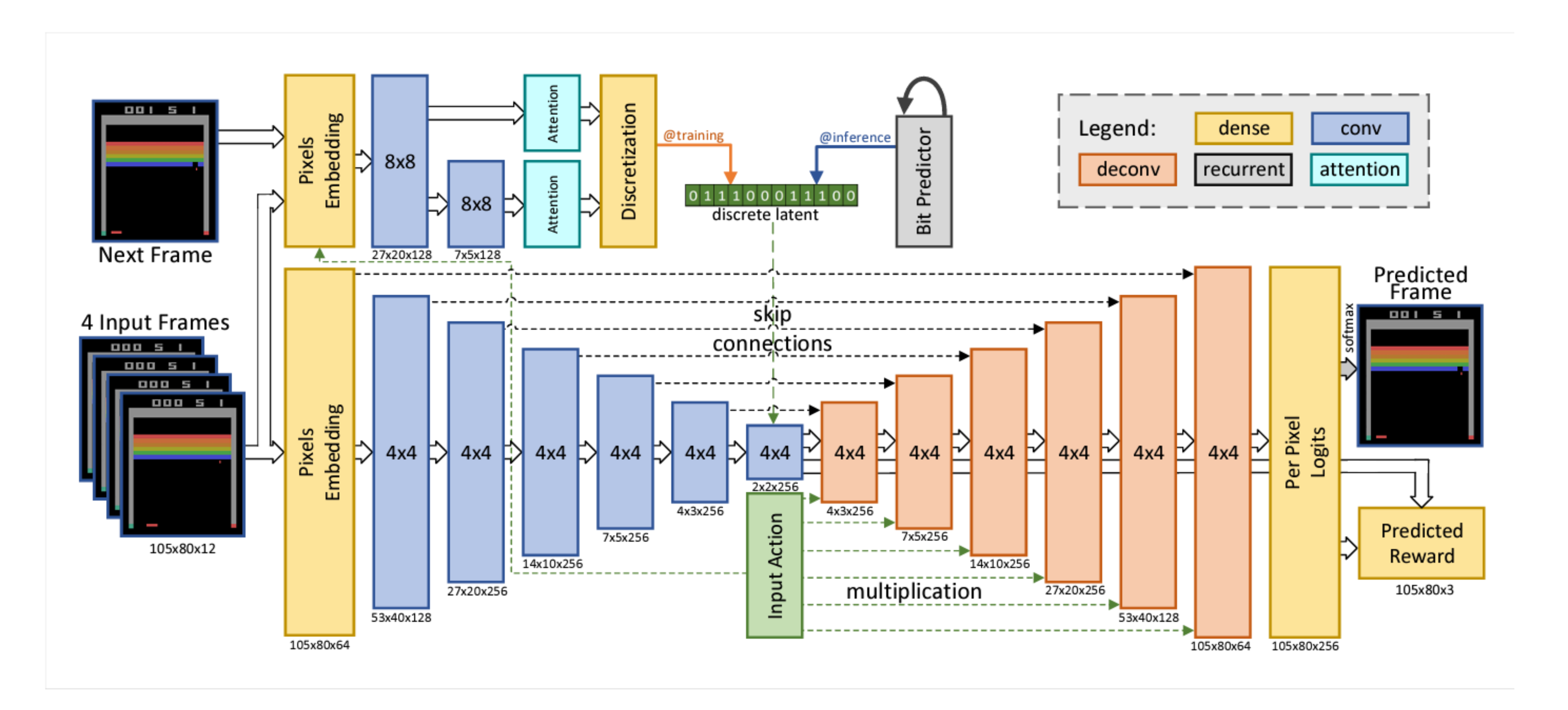


Figure 1: Main loop of SimPLe. 1) the agent starts interacting with the real environment following the latest policy (initialized to random). 2) the collected observations will be used to train (update) the current world model. 3) the agent updates the policy by acting inside the world model. The new policy will be evaluated to measure the performance of the agent as well as collecting more data (back to 1). Note that world model training is self-supervised for the observed states and supervised for the reward.

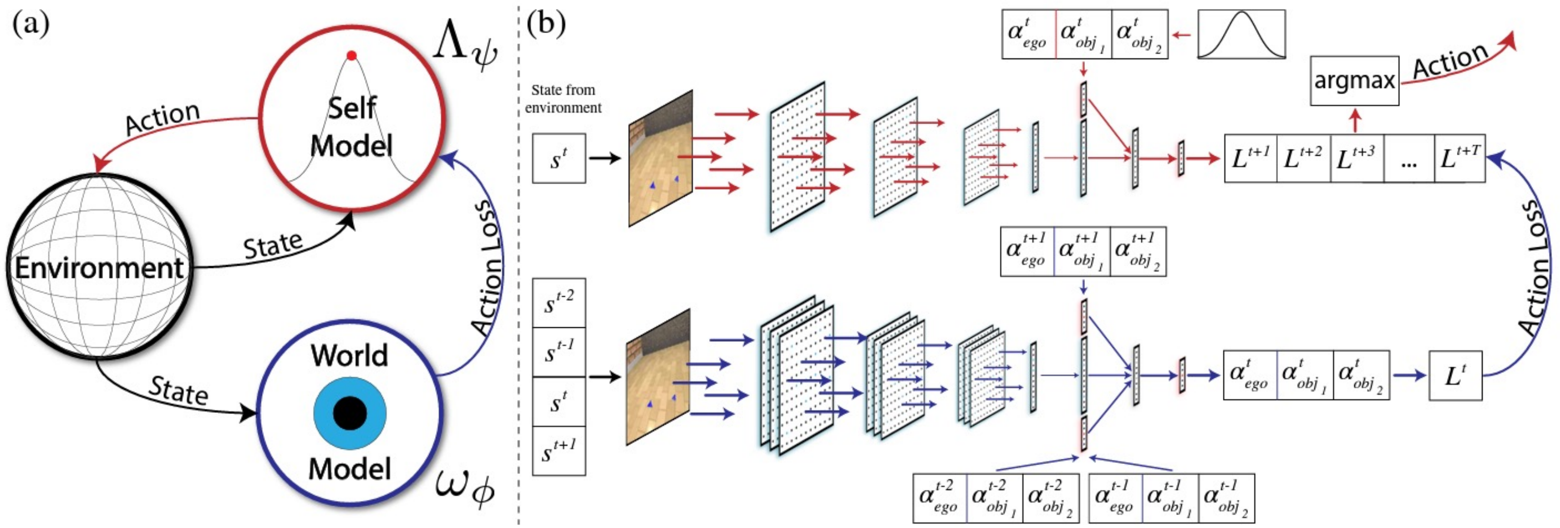
Kaiser et al. “Model-based reinforcement learning for Atari.” ICLR 2020.

“WORLD MODELS”



Kaiser et al. “Model-based reinforcement learning for Atari.” ICLR 2020.

"SELF MODELS"



(a)
Action
Self Model
Λ_ψ
Environment
State
State
World Model
ω_ϕ
Action Loss
(b)
State from environment
s^t
α^t_{ego} $\alpha^t_{obj_1}$ $\alpha^t_{obj_2}$
argmax
Action
L^{t+1} L^{t+2} L^{t+3} ... L^{t+T}
α^{t+1}_{ego} $\alpha^{t+1}_{obj_1}$ $\alpha^{t+1}_{obj_2}$
s^{t-2} s^{t-1} s^t s^{t+1}
α^t_{ego} $\alpha^t_{obj_1}$ $\alpha^t_{obj_2}$
L^t
Action Loss
α^{t-2}_{ego} $\alpha^{t-2}_{obj_1}$ $\alpha^{t-2}_{obj_2}$
α^{t-1}_{ego} $\alpha^{t-1}_{obj_1}$ $\alpha^{t-1}_{obj_2}$

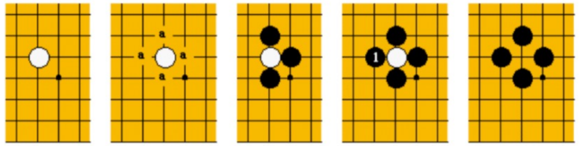
Figure 2: **Intrinsically-motivated self-aware agent architecture.** The world-model (blue) solves a dynamics prediction problem. Simultaneously a self-model (red) seeks to predict the world-model's loss. Actions are chosen to antagonize the world-model, leading to novel and surprising events in the environment (black). (a) Environment-agent loop. (b) Agent information flow.

Haber et al. "Learning to Play With Intrinsically-Motivated, Self-Aware Agents." NeurIPS 2018..

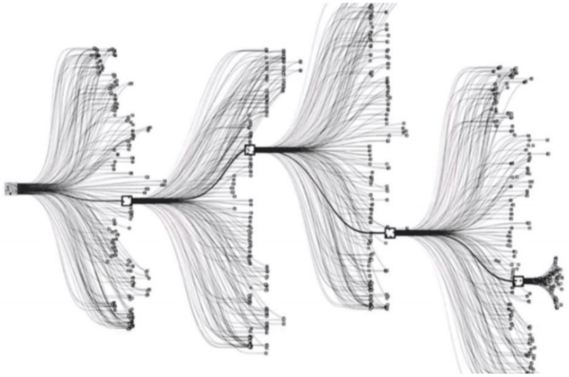
OUTLINE

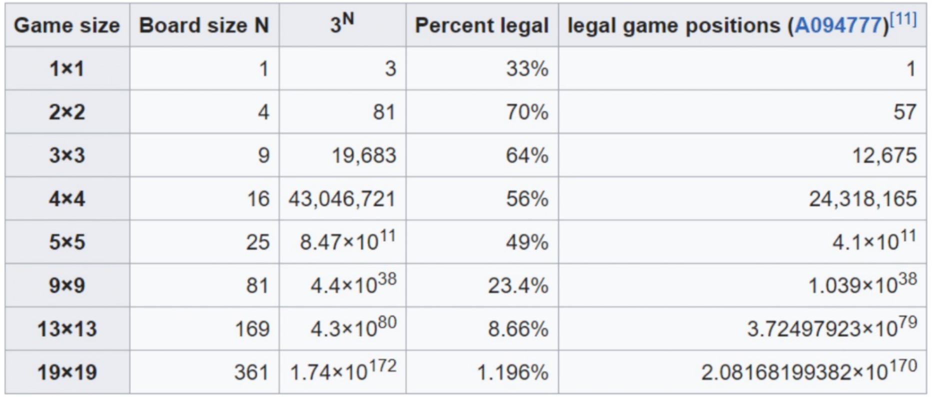
- RL overview
- RL applications
- Deep RL overview
- **Deep RL applications**
- Discussion

GO GAME



Dia. 3 | Dia. 4 liberties | Dia. 5 atari | Dia. 6 capture | Dia. 7 result





Game size	Board size N	3^N	Percent legal	legal game positions (A094777)[11]
1×1	1	3	33%	1
2×2	4	81	70%	57
3×3	9	19,683	64%	12,675
4×4	16	43,046,721	56%	24,318,165
5×5	25	$8.47×10^{11}$	49%	$4.1×10^{11}$
9×9	81	$4.4×10^{38}$	23.4%	$1.039×10^{38}$
13×13	169	$4.3×10^{80}$	8.66%	$3.72497923×10^{79}$
19×19	361	$1.74×10^{172}$	1.196%	$2.08168199382×10^{170}$

Wikipedia

ALPHAGO

Netflix

ALPHAGO

ALPHAGO

- Originally trained by observing historical games by Go experts
- Later, was trained to play against itself using separate instances of itself

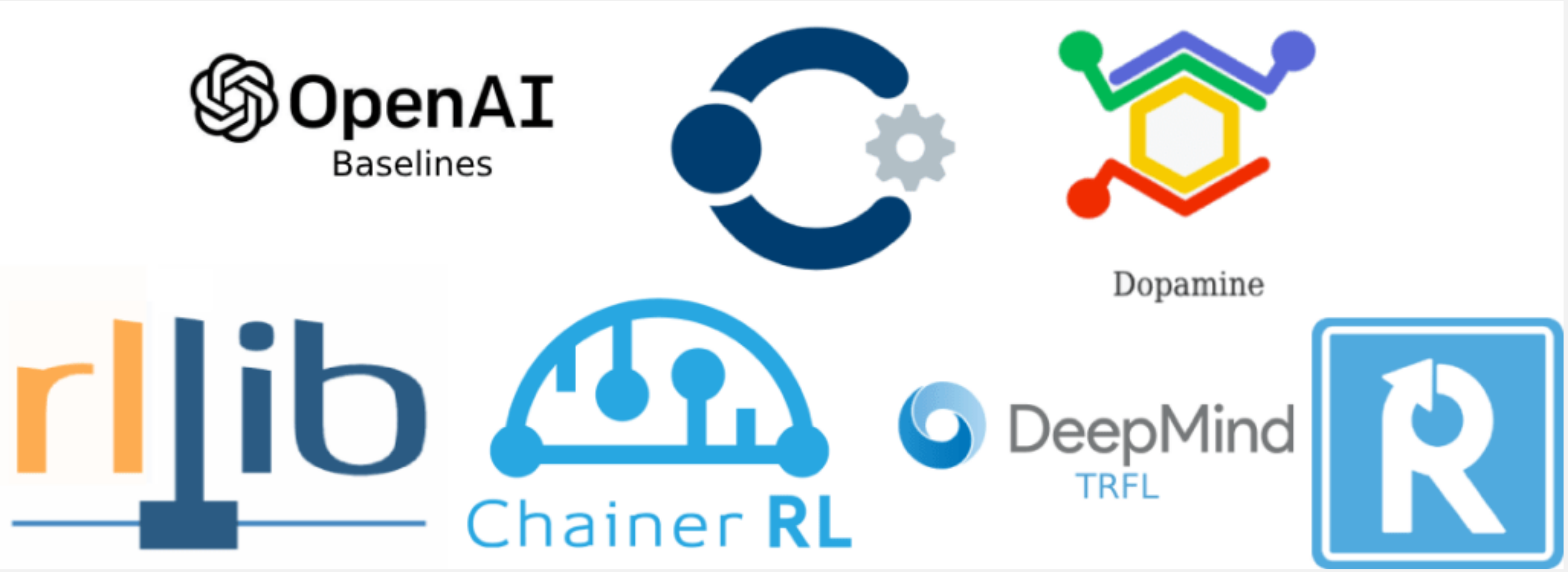
- Wikipedia: "As of 2016, AlphaGo's algorithm uses a combination of machine learning and tree search techniques, combined with extensive training, both from human and computer play. It uses Monte Carlo tree search, guided by a "**value network**" and a "**policy network**," both implemented using **deep neural network** technology. A limited amount of game-specific feature detection pre-processing (for example, to highlight whether a move matches a nakade pattern) is applied to the input before it is sent to the neural networks. The networks are **convolutional neural networks** with 12 layers, trained by **reinforcement learning**."

ALPHASTAR

ALPHAFOLD

- Can accurately predict 3D models of protein structures
- Has the potential to accelerate research in every field of biology

IMPLEMENTING RL IN PYTHON



OUTLINE

- RL overview
- RL applications
- Deep RL overview
- Deep RL applications
- Discussion

DISCUSSION QUESTIONS

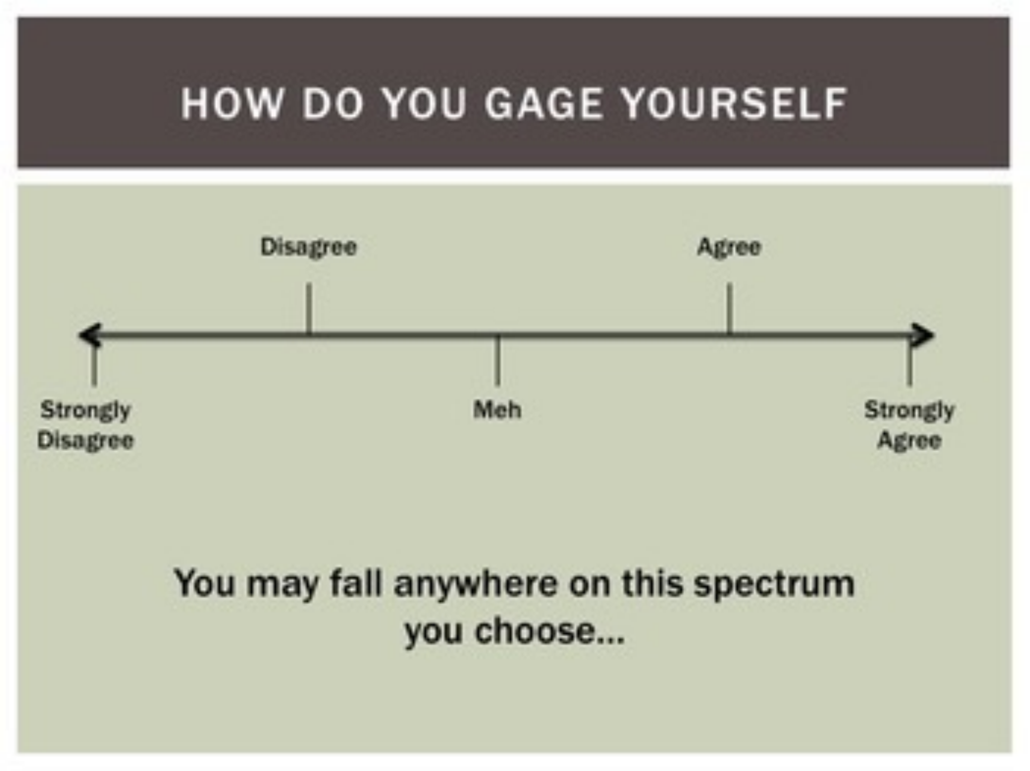
Split into 4 groups of 5 and discuss one of the following topics for 10-15 minutes:

1. Deep RL is one creative method of integrating several independent deep networks into a single system. What are ideas for other creative systems which combine 2 or more deep networks into a single system?
2. Who should design Reward functions in real-world RL systems? Should policy makers have a say? How can the design of Reward functions be made transparent to non-technical stakeholders?
3. David Silver (lead of AlphaGo team) observed that AlphaGo played most of a game seeming to falsely think it was winning but was actually losing. He claims that "you have to trust in them [RL] to know better than you, the designer." Discuss.
4. There are at least 2 large paradigms for training RL agents: (1) real-world observation followed by one-shot trial-and-error, and (2) realistic simulation followed by transfer learning. What are important considerations when using either method?

Present your consensus, or the differing opinions in the group, to the class. All group members should present.

DISCUSSION QUESTIONS (CONTINUED)

Statement: Reinforcement Learning is the right approach to take when working towards Artificial General Intellgience.


HOW DO YOU GAGE YOURSELF
Disagree
Agree
Strongly Disagree
Meh
Strongly Agree
You may fall anywhere on this spectrum you choose...