WEEK02 : CS234 Lecture 01

Lecture Link

https://www.youtube.com/watch?v=FgzM3zpZ55o&list=PLoROMvodv4rOSOPzutgyCTapiGlY2Nd8u

Slides

https://web.stanford.edu/class/cs234/CS234Win2019/schedule.html

Reinforcement Learning

Today's Goal

- Overview of RL
- Course Logistics
- Introduction to Sequential Decision Making under uncetainty

Overview of RL

"How can an Intelligent Agent learn to make a good sequence of decisions?"

- AI 모델이 어떻게 하면 연속적인 결과를 잘 만들어낼 수 있도록 학습시킬 수 있을까?

⇒ 이 대답이 RL

Sequences of Decisions

어떤 Intelligent Agent가 하나의 결정이 아닌 Sequence of Decision을 내림

Good

어떻게 최적 알고리즘을 적용하는가

- RL의 특징은 “학습"에 있다.
- 현재는 내 결정이 world에 어떤 영향을 끼칠지 모름

Learn

- 경험을 통한 정보가 있어야 함

Fundamental Challenge

AI and ML은 불확실성속에서 좋은 결정을 만들기 위해 학습한다.

Figure: Example from Yael Niv

- 살아가며 성장함
  - 아기 : 원시 뇌 + 눈 한개 ⇒ 돌에 붙음
  - 어른 : 뇌를 소화하고 정착함
- 뇌를 가지고있다 = decision을 guide하는데 도움이 된다.
  - life가 완성된 agent는 뇌가 필요없다 (?)
- 왜 “agent”가 똑똑해야 하는가? or 왜 “agent”가 결정을 내리는가
- RL은 많은 분야에 적용 가능
  - 기회이자 책임을 지녀야 함.

ATATRI

Agent가 pixel input을 가지고 바로 학습함

→ 색칠해진 픽셀 Input

Robotics

Berkely → grasping, cloths, 등의 연구

Educational Games

- how AI to help amplify human potential?
  - goal = 사람들에게 “분수"등과 같은 개념을 빠르고 효과적으로 가르칠 필요가 있었음.
- AI가 인간의 잠재력을 늘리는데 기여
  - Educational Game

Healthcare

RL을 EMR이나 환자 기록을 가지고 어떻게 하는가?>

NLP, Vision, ...

optimization problem을 많이 해소중

RL Involves

요약

- optimization
- Delayed Consequences

- Exploration
- Generalization

Optimization

- 목표 : 의사결정을 만드는데 최적의 방법 or 전략 찾기

Delayed Consequences

- 예시
  - 지금 한 행동이 많이 지난 후에 영향을 끼치기도 함
  - 게임에서 초반에 나오는 아이템이 나중에 쓰이기도 함
  - 오늘 공부를 열심히하면 3주뒤 중간고사를 잘 봄
- 즉, 과거의 데이터가 미래에 영향을 줄 수 있음

Exploration

- 의사결정을 통해 세상을 알아감
  - Agent를 과학자라고 하면
  - 자전거 타기에 실패하면서, 어떻게 타는지를 분석
  - 그러면 중력과 물리학에 대한 개념을 알게됨
- Censored Data
  - Data에서 “보상"만 얻음
    - 하고싶은거만 배우려고 함

Generalization

의문점

- 정책은 과거 행동으로 매핑

- pre-program policy가 아닌가?

정책

- 과거 경험을 행동과 매핑
- 사전에 프로그램을 만들어두며 되는거 아닌가?

⇒ 미리 만들어둔 정책은 새롭게 나타난 배열에 대해서 예측하지 못할 것이다.

AI Planning vs RL

- Planning Game : 아래의 3가지를 포함
  - Optimization
  - Generalization
  - Delayed Consequences
- ex) 바둑 : 바둑돌을 움직임
  - 초반에 움직인 돌에 대한 즉각적 반응이 없음
  - 나중에야 그 수가 좋은 수였는지를 알 수 있다.
- Idea & Planning : 어떻게 세상이 돌아가는지를 모델에게 알려주자.
  - Reward가 무엇인가?
  - 세상이 돌아가는걸 계산하는 방법을 알려줘야 함
  - 그래서 “exploaration”은 필요가 없다.

Supervised Learning vs RL

- Supervised Learning
  - Optimization
  - Generalization
  - No Exploration

        - 이미 Dataset이 주어짐
        - 세상속에서 데이터를 우리가 수집할 필요가 없다.
    - No Delayed Consequences
        - 바로 정답 or Not으로 나뉨
- 특징
    - 경험을 통해 학습함
    - 세상은 Label이 없다.
        - 우리가 직접 Label을 만들어줘야 함.
        - 그러나 이 Label이 진정한 Label인지를 모른다.

Imitation Learning vs RL

- Imitation Learning
    - Optimization
    - Generalization
    - No Exploration
    - Delayed Consequences
- 특징
    - 다른 사람 or 무언가의 의사결정 과정을 보고 따라하는 연구
    - 경험을 통해 학습하는 다른 분야
    - Input이 좋은 정책이라는 가정
- Example : 장난감 헬리콥터를 날리는 연구
    - 하나의 행동만 학습하면
      이전에 보지 못한 행동이 나왔을 때 대처하지 못한다.

어떻게 하면 좋을까?

위에 나온 4가지를 다 사용할 줄 알아야 한다.

- Explore the world
- Use experience to guide future decisions

Other Issues

- “보상"은 어디서 나오는가?
  - 이걸 잘못 설정하면 어떻게 되는가?
  - 이게 좋은 결정인지 아닌지를 결정해야하는데, Setting이 잘못되면 어떻게 해야하는가.
- Robustness / Risk Sensitivity
- We are not alone..
  - Multi - agent RL

Course Logistics

선수 과목

- 파이썬
- 기초 통계학
- 다변량 미적분 & 선형대수
- Machine Learning for AI
- Loss Functions, 편미분, Gradient Descent
- 마르코프 의사결정은 다룰 예정

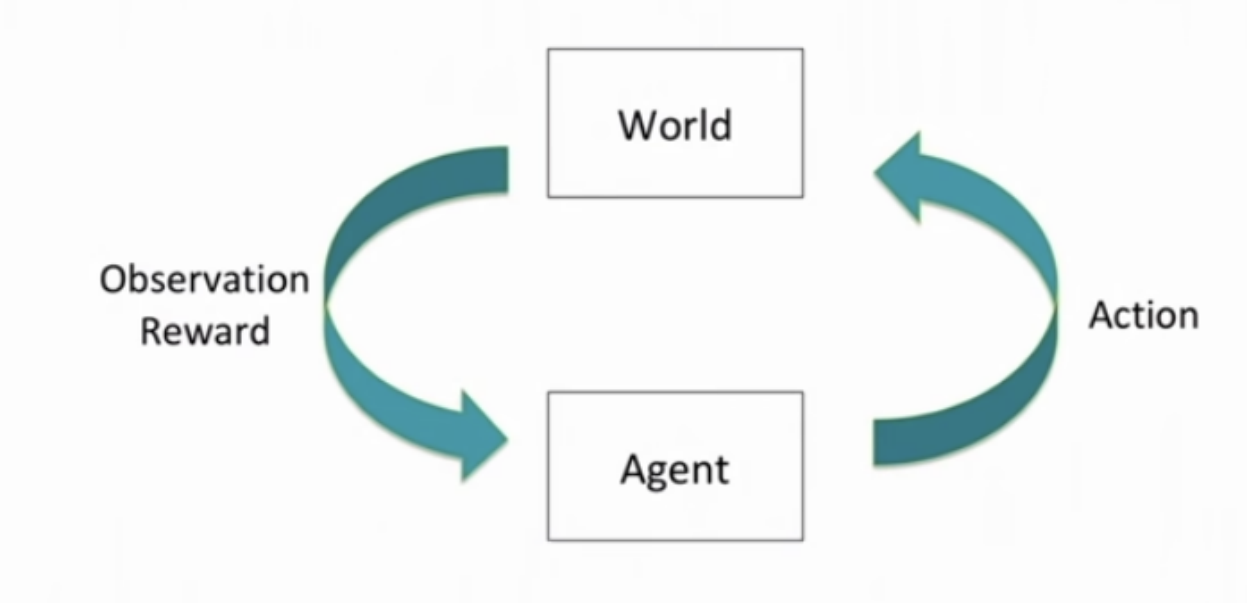
수업 목표

- RL의 주요 Feature를 정의하고, AI나 다른 ML과 분리해내야 함
- 기존 연구 분야를 RL 수식으로 재정립 (수식은 뭐고 ~~)
- RL알고리즘 구현
- RL 알고리즘 분석으로 다양한 기준을 설명 & 평가
- exploration vs exploitation의 차이를 설명하고
  비교 대조 할 수 있어야 함

Introduction to Sequential Decision Making under Uncertainty

Sequential Decision Making

개념



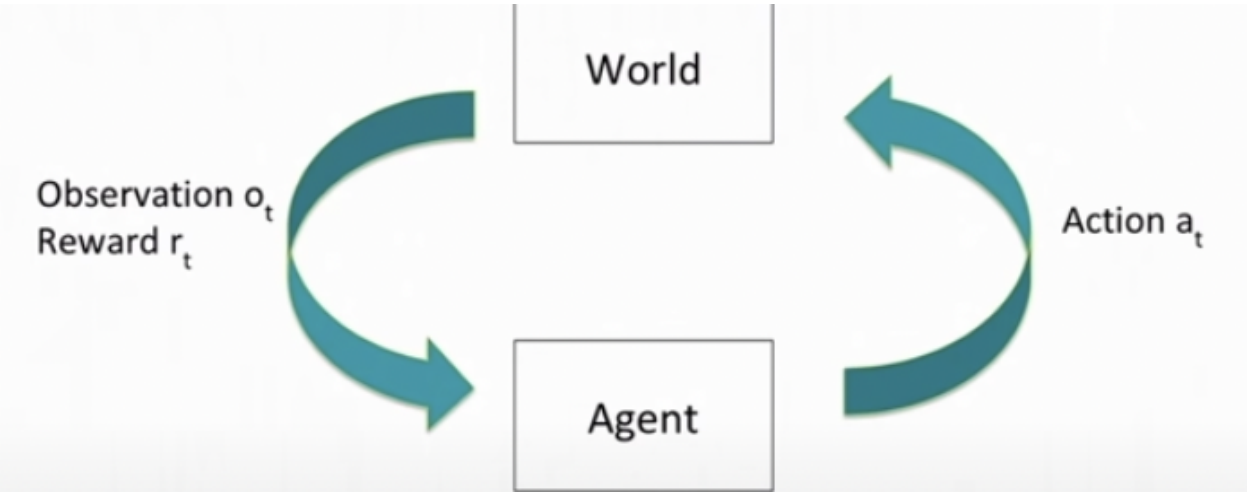
세상 → 관찰 → Agent → Action → 세상

- 목표 : 최종 Reward를 최대화 하는 방향으로 행동하자
  - 즉각적 Reward와 Long-term reward

Example

- 웹 광고
  - World → 웹사이트 체류 시간 & 광고 클릭 → Agent → 광고 설정
- Robot Unloading Dishwasher
  - World → 부엌 카메라 : no dishes면 reward → Agent → Move
- Tutoring
  - 더하기 빼기도 모르는 학생들
  - 더하기 빼기를 가르쳐주는 Agent
  - Agent는 문제를 맞추면 +1, 틀리면 -1
  - 이 Reward를 가장 늘리기 위한 Optimization?

과거



실제 세계

- Agent에게 Observation이 숨어있을 수 있다.
  - 누가 뒤에서 내 머리 치려는건 관찰이 안됨

Markov 가정

특징

- 현재는 과거를 담고있기에, 미래를 예측하는데는 현재만있으면 된다.
- $p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t)$
- 미래는 과거와 독립

왜 유명한가?

- 언제나 만족됨 ($s_t = h_t$)를 항상 만족할 수 있음
- 가장 최근의 observation이 과거의 정보를 담고있다는 가정 ($s_t = o_t$)도 많음
- State Representation이 함축적으로 담은 의미
  - 계산 가능한 복잡성
  - Data Required
  - Resulting Performance

Partial Observability / Partially Observable MDP

- Agent의 상태는 매번 달라짐
- Agent가 자신의 상태를 판단할 때
  - $s_t = h_t$, word state, RNN 등을 활용

예시 : 포커

- Player : 자신의 카드만 봄
- Healthcare : 모든 심리학적 과정

Bandit

- 매우 쉬운 version of MDP
- No delayed rewards
  - 현재의 행동이 다음의 관찰에 영향을 주지 않는다고 가정
- Example) 웹페이지 광고의 경우,
  - User 1의 광고가 User 2의 광고 설정에 영향을 주지 않음
  - 서로 독립을 가정

MDP & POMDP

닫힌 구조를 생각해야 한다.

- 현재의 Action이 세상을 바꾸는 중이다.
- Action → Change State → ...
- 내 의견이 다음 사람에게 영향을 줄 것이다.

결정론적 변화

- 다른 상태로 이동 → 결정론적 변화
- robotics and controls
- Example
  - 동전을 던지기 → 결정론적 과정임에도 통계학적으로 바라본다. (p=0.5)
- 충분히 완벽한 모델이 있다면, 결정론적으로 보일 것이다.

Example : Mars Rover as PMDP

화성 탐사 로봇

- 방금 막 착륙
- 7개의 State가 있다고 가정

- Action
  - 좌측, 우측 중 고민중
- Rewards
  - +1 in state s_1
  - +10 in state $s_7$0 in all other states
- 즉, 최초의 선택 방향이 미치는 영향으로 인해, 최후에 어떤 영향을 끼쳤는지를 보자.

RL Algorithm Components

요약

주로 아래의 3개중 하나 이상은 반영함

- Model
  - world의 변화를 나타내어 agent의 action에 반응
- Policy
  - function mapping agent’s states to action
- Value Function
  - 미래의 Reward

Model

- Agent’s Representation
  - Agent의 행동으로 인해 세상이 어떻게 변했는지를 나타냄
- Transition / dynamics model이 다음 상태 예측
  - $p(s_{t+1} = s' | s_t = s, a_t = a)$
- Reward model : 즉각적 보상 예측

- $r(s_t = s, a_t = a) = E[r_t | s_t = s, a_t = a]$

예시

s_1	s_2	s_3	s_4	s_5	s_6	s_7
$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$

- 모든 보상이 0
- motor control을 매우 못함
- 50%의 확률로 남아있거나 움직임
  - 0.5 = $P(s_1 | s_1, TryRight) = P(s_2 | s_1, TryRight)$
  - 0.5 = $P(s_2 | s_2, TryRight) = P(s_3 | s_2, TryRight)$
  - ,,,
- 모델이 틀릴 수 있지만, agent에게 useful할 수 있음

Policy

- 정책 π : agent의 행동을 선택
- $\pi : S \rightarrow A$, state에 따른 행동 선택
- 결정론적 정책
  - $\pi(s) = a$
- 확률론적 정책 : action에 대한 distribution을 이용
  - $\pi(a|s) = Pr(a_t = a | s_t = s)$
  - 동전을 던져서 앞면이면 공항으로, 뒷면이면 기차역으로 가자

Value Function

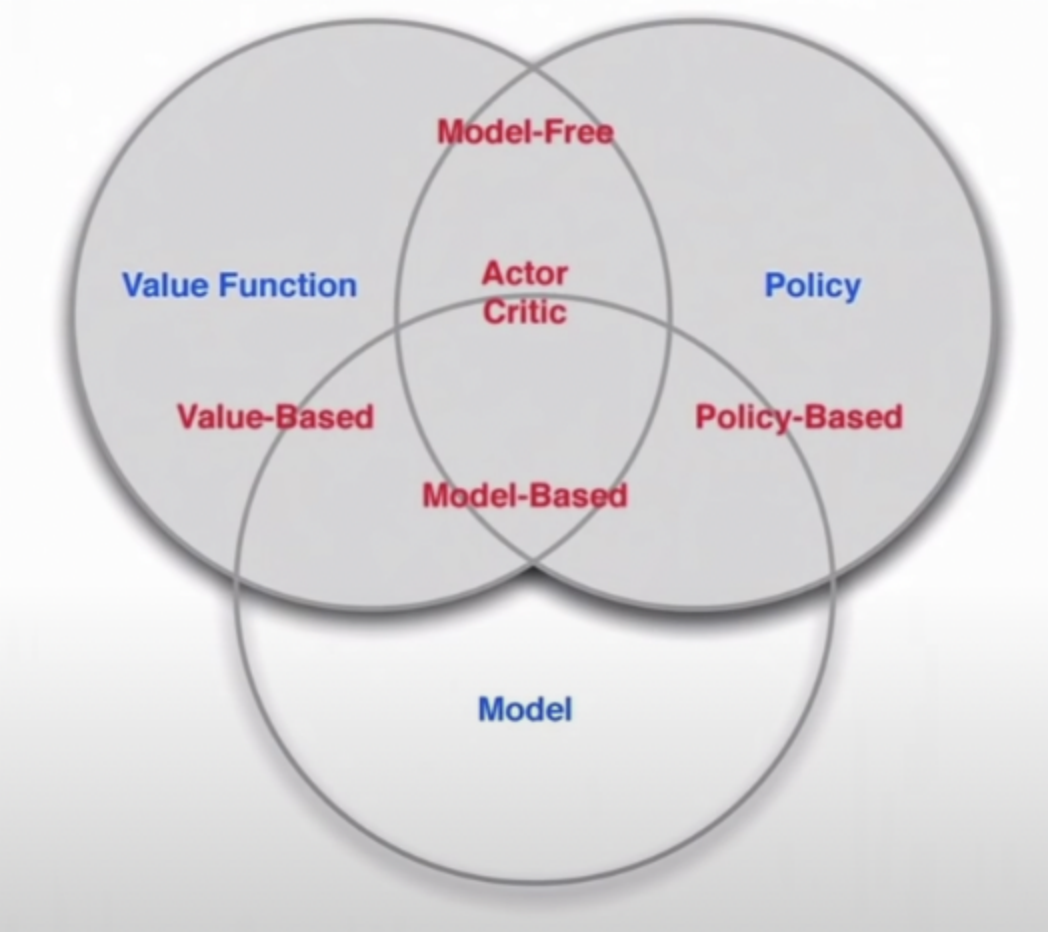
- discounted sum of future rewards under a particular policy π
- 특정 정책 π를 따를 때의 미래의 reward

$$V^{\pi}(s_t = s) = \mathbb{E}_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

- γ : 할인 요소 : 즉각적이냐 or 미래의 관점이냐에 차이
  - 0이면 즉각적 반응
  - 1이면 미래만 봄
- 현재 행동의 정도를 추정
- 정책을 비교하면서 결정가능

Types of RL Agents

- Model - based
  - Explicit : Model
  - Policy나 Value Function이 없을 수 있음
- Model-free
  - Explicit : Value Funtion and or policy function
  - no model



Key Challenges in Learning to Make Sequences of Good Decisions

- Planning : Agent의 내부 계산
  - Given Model of how the world works
    - Dynamics and reward model
  - Algorithm computes how to act in order to maximize expected reward
    - with no interaction with real environment
  - 게임의 규칙을 알고 있을 때 많이 사용
    - 상태 s에 따른 행동 a
    - 다음 상태에 대한 분포 계산이 가능
    - 예상 점수 계산 가능

        - optimal action을 결정 가능
- RL
    - Agent가 world를 모름
    - Interacts with world do implicitly or explicitly learn how world works
    - Agent Improves policy
    - 규칙을 몰라도 적용 가능
        - 행동과 결과로 학습
        - 좋은 정책을 학습하려 함

Exploration and Exploitation

- Agent는 예전에 일어났던 행동을 결정하려 함
- RL agent가 어떻게 action에 balance를 맞추는가?
    - Exploration : 미래에 더 좋은 결정을 하기 위한 탐색 (새로운 행동도 시행)
    - Exploitation : 과거 경험을 바탕으로 좋은 결과를 얻기 위한 행동 선택
    - 주로 이 둘 간의 trade-off가 발생
        - reward를 희생해서 탐색 or 더 좋은 정책 탐색
- 예시
    - 영화
        - Exploitation : 좋아하는 영화 보기
        - Exploration : 새로운 영화 보기
    - 광고
        - 최적 광고 설정
        - 다른 광고 설정
    - 운전
        - 최적 경로 설정

    - 다른 루트 설정

평가 & 조종

- Evaluation : 미래의 Reward 추정
- Contorl : 최적 정책 찾기