화학과 강화학습 세미나

임도형
dh-rim@hanmail.net

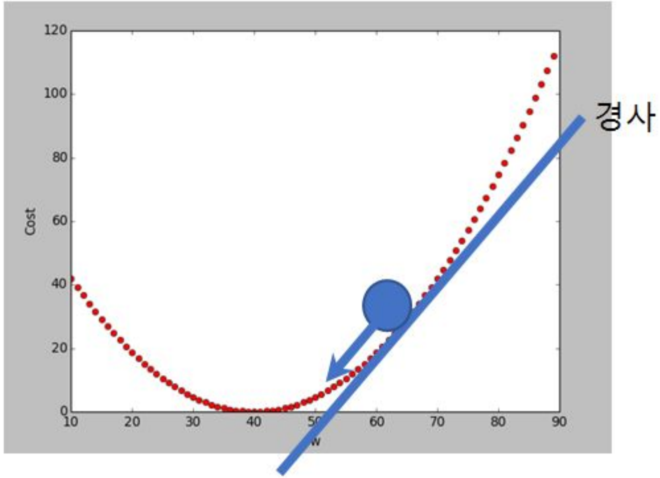
DNN

DNN is Universal Approximator

- DNN은 임의의 함수를 근사화 할 수 있다.
- 함수의 내부를 모르더라도
- 입력과 출력의 쌍으로

Gradient Descent

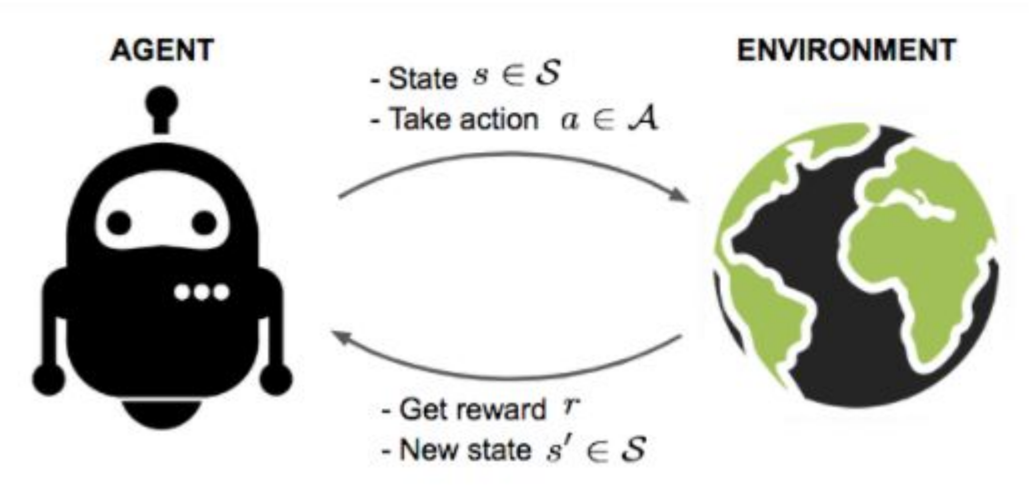
- cost function을 정의하고
- DNN을 정의하는 weight들에 의한 cost function의 경사를 구한다.
- 그리고 그 경사를 사용하여 weight들을 업데이트 한다.



$$\theta = \theta - \alpha \frac{dLoss}{d\theta}$$

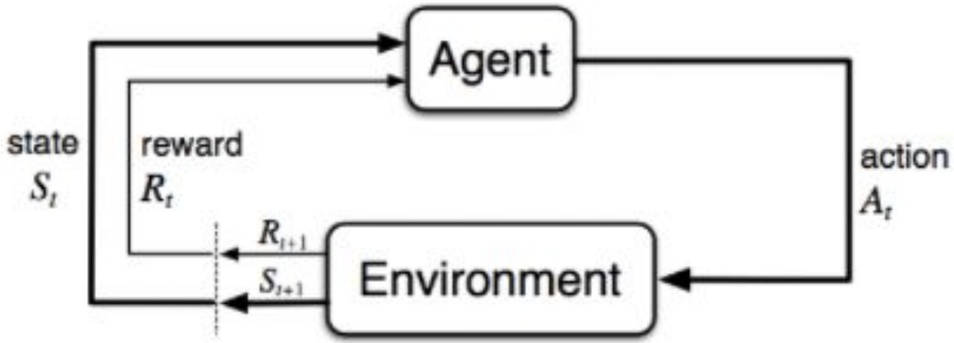
고전적인 강화학습

Agent and Environment



https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html#policy-gradient

State, Action, Reward



https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html#policy-gradient

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

θ : pi를 정의하는 파라매터

$d^\pi(s)$ is the stationary distribution of Markov chain for π_θ

$V(s)$	State-value function measures the expected return of state s; $V_w(.)$ is a value function parameterized by w.
$V^\pi(s)$	The value of state s when we follow a policy π; $V^\pi(s) = \mathbb{E}_{a \sim \pi}[G_t \mid S_t = s]$.
$Q(s, a)$	Action-value function is similar to $V(s)$, but it assesses the expected return of a pair of state and action (s, a); $Q_w(.)$ is a action value function parameterized by w.
$Q^\pi(s, a)$	Similar to $V^\pi(.)$, the value of (state, action) pair when we follow a policy π; $Q^\pi(s, a) = \mathbb{E}_{a \sim \pi}[G_t \mid S_t = s, A_t = a]$.

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a)$$

폴리시 함수 pi에 의해
상태 s일때 액션 a를 할때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s,a)$$

폴리시 함수 pi에 의해
상태 s일때 액션 a를 할때의 가치

폴리시 함수 pi에 의해
상태 s일때 액션 a가 발생할 확율

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a)$$

폴리시 함수 pi에 의해
X 상태 s일때 액션 a를 할때의 가치

폴리시 함수 pi에 의해
상태 s일때 액션 a가 발생할 확율

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

상태 s일때
폴리시 함수 pi에 의해
액션 a를 할때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

모든 액션에 대하여

상태 s일때
폴리시 함수 pi에 의해
액션 a를 할때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \underline{\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a)}$$

모든 액션에 대하여
상태 s일때
폴리시 함수 pi에 의해
액션 a를 할때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

상태 s일때
폴리시 함수 pi에 의한 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \underline{\sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s,a)}$$

폴리시 함수 pi에 의해
상태 s일 때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) \underline{V^\pi(s)} = \sum_{s \in \mathcal{S}} d^\pi(s) \underline{\sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)}$$

폴리시 함수 pi에 의해
상태 s일 때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} \underline{d^\pi(s)}\,\underline{V^\pi(s)} = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

폴리시 함수 pi에 의해
상태 s가 발생할 확률

폴리시 함수 pi에 의해
상태 s일 때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} \underline{d^{\pi}(s) V^{\pi}(s)} = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

폴리시 함수 pi에 의해
상태 s가 발생할 확률 x 상태 s일 때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} \underline{d^{\pi}(s) V^{\pi}(s)} = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

폴리시 함수 pi에 의해
상태 s가 발생할 때의 가치

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

모든 상태에 대하여

폴리시 함수 pi에 의해
상태 s가 발생할 때의 가치

Reward Function

$$J(\theta) = \underline{\sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s)} = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

폴리시 함수 pi에 의해
모든 상태에 대하여
각 상태 s가 발생할 때의 가치의 합

Reward Function

$$J(\theta) = \underline{\sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s)} = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

폴리시 함수 pi에 의해
모든 상태에 대한 가치의 합

Reward Function

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

폴리시 함수 pi에 의한 가치

Reward Function

$$J(\theta) = \underbrace{\sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s)} = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s,a)$$

폴리시 함수 pi에 의한 가치

강화학습의 목적

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

- 보상을 최대로 하는 pi를 구하자.
- pi를 정의하는 theta를 찾자.

Value Function

$$\begin{aligned} V(s) &= \mathbb{E}[G_t | S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) | S_t = s] \end{aligned}$$

Action-Value Function, Q-value

$$
\begin{aligned}
Q(s,a) &= \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \mathbb{E}[R_{t+1} + \gamma \mathbb{E}_{a\sim\pi} Q(S_{t+1}, a) \mid S_t = s, A_t = a]
\end{aligned}
$$

Value Function, Action-Value Function

$$V(s) = \mathbb{E}[G_t | S_t = s]$$
$$= \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) | S_t = s]$$

$$\begin{aligned} Q(s,a) &= \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \mathbb{E}[R_{t+1} + \gamma \mathbb{E}_{a \sim \pi} Q(S_{t+1}, a) \mid S_t = s, A_t = a] \end{aligned}$$

Bellman Optimality Equations

value function, action-value function을 최대화하는 pi

$$V_*(s) = \max_\pi V_\pi(s)$$

$$Q_*(s,a) = \max_\pi Q_\pi(s,a)$$

$$\pi_* = \arg\max_\pi V_\pi(s)$$

$$\pi_* = \arg\max_\pi Q_\pi(s,a)$$

강화학습

Bellam Optimality Equation의 해를 푸는 과정

$$V_*(s) = \max_{\pi} V_{\pi}(s)$$

$$Q_*(s,a) = \max_{\pi} Q_{\pi}(s,a)$$

$$\pi_* = \arg\max_{\pi} V_{\pi}(s)$$

$$\pi_* = \arg\max_{\pi} Q_{\pi}(s,a)$$

강화학습의 방법들

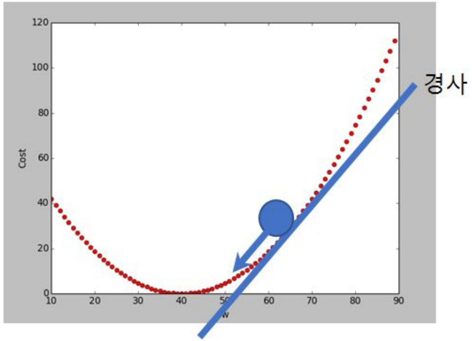
$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s, a)$$

- Dynamic Programming
- Monte-Carlo methods
- Temporal-Difference Learning
- MC와 TD 결합
- Evolution Strategies
- Policy Gradient

현재의 강화학습

Gradient Descent

$$\theta = \theta - \alpha \frac{dLoss}{d\theta}$$



Gradient Descent 적용

$$J(\theta) = \sum_{s \in \mathcal{S}} d^{\pi}(s) V^{\pi}(s) = \sum_{s \in \mathcal{S}} d^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s, a)$$

- reward function은 cost function의 반대이다. 클수록 **<-->** 작을수록 좋다.
- $\frac{d}{d\theta} J(\theta)$, 즉 $\nabla_{\theta} J(\theta)$ 을 구해 반대로 더하자.

Policy Gradient Theorem

$$J(\theta) = \sum_{s \in \mathcal{S}} d^\pi(s) V^\pi(s) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a)$$

- reward function은 cost function의 반대이다.
- $\frac{d}{d\theta} J(\theta)$, 즉 $\nabla_\theta J(\theta)$ 을 구해 반대로 더하자.

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \pi_\theta(a|s) \\
&\propto \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s) \\
&= \mathbb{E}^\pi \left[\nabla_\theta \pi(s,a)\, Q^\pi(s,a) \right]
\end{aligned}$$

REINFORCE

1. Initialize the policy parameter θ at random.
2. Generate one trajectory on policy π_θ: $S_1, A_1, R_2, S_2, A_2, \ldots, S_T$.
3. For t=1, 2, ... , T:
    1. Estimate the the return G_t;
    2. Update policy parameters: $\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$$

$s \in \mathcal{S}$	States.
$a \in \mathcal{A}$	Actions.
$r \in \mathcal{R}$	Rewards.
G_t	Return; or discounted future reward; $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$
$\pi(a\|s)$	Stochastic policy (agent behavior strategy); $\pi_\theta(.)$ is a policy parameterized by θ.

REINFORCE

$$\theta \leftarrow \theta + \underline{\alpha} \gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$$

학습율.
대략 0.0001

REINFORCE

$$\theta \leftarrow \theta + \alpha\underline{\gamma^t} G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$$

감쇄율.
대략 0.95

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t \underline{G_t} \nabla_\theta \ln \pi_\theta(A_t|S_t)$$

time t일 때의 가치

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \underline{\pi_\theta(A_t|S_t)}$$

time t일 때의 액션 확률

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t G_t \underbrace{\nabla_\theta \ln \pi_\theta(A_t|S_t)}_{}$$

time t일 때의 액션 확률의 gradient

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$$

time t일 때의 가치

time t일 때의 액션 확률

REINFORCE

$$\theta \leftarrow \theta + \alpha\gamma^t \underbrace{G_t}_{\text{time t일 때의 가치}} \nabla_\theta \underbrace{\ln \pi_\theta(A_t|S_t)}_{\text{time t일 때의 액션 확율}}$$

- 2개의 모델
  - Policy Network : a = P(s). time t일 때의 액션을 출력. 모델의 weight가 theta
  - Value Network : v = V(s). time t일 때의 가치를 출력

REINFORCE - 실제 적용

$$\theta \leftarrow \theta + \alpha\gamma^t \underbrace{G_t}_{\text{Value Network}} \nabla_\theta \underbrace{\ln \pi_\theta(A_t|S_t)}_{\text{Policy Network}}$$

- Stage 1. Value Network을 먼저 학습시킨다.
- Stage 2. Value Network을 가지고 Policy Network을 학습 시킨다.

Actor-Critic

$$\nabla_\theta J(\theta) = \nabla_\theta \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} Q^\pi(s,a)\pi_\theta(a|s)$$
$$\propto \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} Q^\pi(s,a)\nabla_\theta\pi_\theta(a|s)$$
$$= \mathbb{E}^\pi \left[\nabla_\theta \pi(s,a)\ Q^\pi(s,a)\right]$$

- 2개의 모델
  - Actor : policy parameter theta를 업데이트
  - Critic : value function parameter w를 업데이트.

Actor-Critic

1. Initialize s, θ, w at random; sample $a \sim \pi_\theta(a|s)$.
2. For $t = 1 \dots T$:
    1. Sample reward $r_t \sim R(s, a)$ and next state $s' \sim P(s'|s, a)$;
    2. Then sample the next action $a' \sim \pi_\theta(a'|s')$;
    3. Update the policy parameters: $\theta \leftarrow \theta + \alpha_\theta Q_w(s, a) \nabla_\theta \ln \pi_\theta(a|s)$;
    4. Compute the correction (TD error) for action-value at time t:
       $\delta_t = r_t + \gamma Q_w(s', a') - Q_w(s, a)$
       and use it to update the parameters of action-value function:
       $w \leftarrow w + \alpha_w \delta_t \nabla_w Q_w(s, a)$
    5. Update $a \leftarrow a'$ and $s \leftarrow s'$.

with environment - 실제 적용

$$\theta \leftarrow \theta + \alpha\gamma^t \underbrace{G_t}_{\text{Environment}} \nabla_\theta \ln \underbrace{\pi_\theta(A_t|S_t)}_{\text{Policy Network}}$$

- Environment에서 reward를 구하고 이를 가지고 Policy Network을 학습 시킨다.
- Atari Game등의 경우.

Policy Gradient Algorithms

- REINFORCE
- Actor-Critics
- Off-Policy Policy Gradient
- A3C
- A2C
- DPG
- D4PG
- MADDPG
- TRPO
- PPO
- PPG
- ACER
- ACTKR
- SAC
- SAC with Automatically Adjusted Temperature
- TD3
- SVPG
- IMPALA

Reinforcement Learning vs REINFORCE

- 혼동되는 용어
- 강화학습 vs 강화학습?
- 기존 강화학습에 DNN을 적용하니 잘 풀린다.
- DNN으로 Policy Function을 구성하고 Gradient를 사용하여 학습.
- 이제는 전통적인 방법을 대신하여 DNN을 사용한 것을 강화학습이라 칭한다.

보다 쉬운 강화학습

Value Network as Reward Function

$$\theta = \theta - \alpha \frac{dLoss}{d\theta}$$

- 전통 강화학습과 관계 없다.
- - Value Network을 Loss Function으로 한다.
- 이 Loss Function으로 대상 네트웍을 직접 학습 시킨다.

$$\theta = \theta - \alpha \frac{d(-V)}{d\theta} = \theta + \alpha \frac{dV}{d\theta}$$

Reference

- A (Long) Peek into Reinforcement Learning : https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html#key-concepts
- Policy Gradient Algorithms : https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html
- Reinforcement Learning : An Introduction : https://www.slideshare.net/carpedm20/reinforcement-learning-an-introduction-64037079