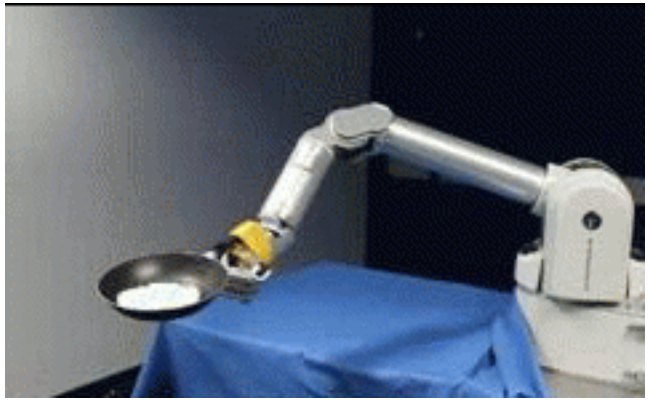
OpenAI Gym ve Python ile Pekiştirmeli Öğrenmeye Giriş

Cem Eteke

ceteke13@ku.edu.tr

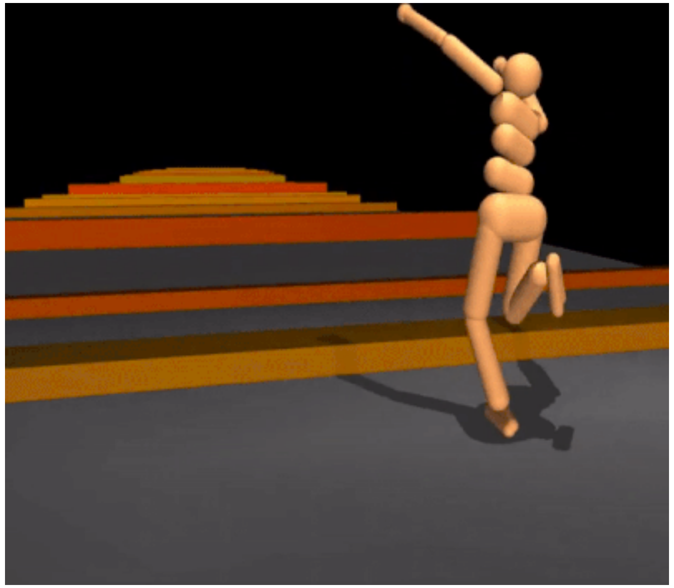
Taslak

- Pekiştirmeli öğrenmeye giriş
- OpenAI Gym
- Model tabanlı öğrenme
  - Jupyter Notebook örneği
- Modelsiz öğrenme
- Yaklaşık Öğrenim
  - Jupyter Notebook örneği
- Politika tabanlı öğrenme
  - Jupyter Notebook örneği



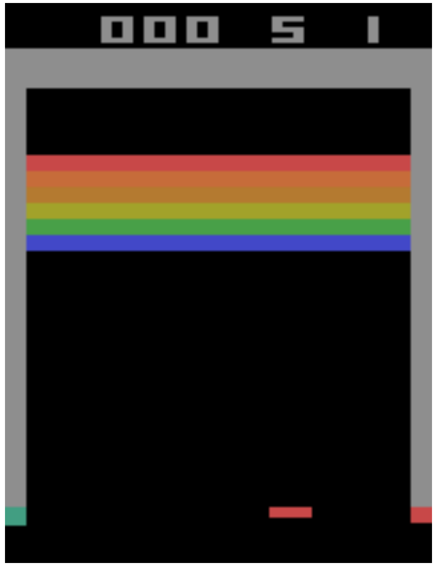
Pekiştirmeli Öğrenme

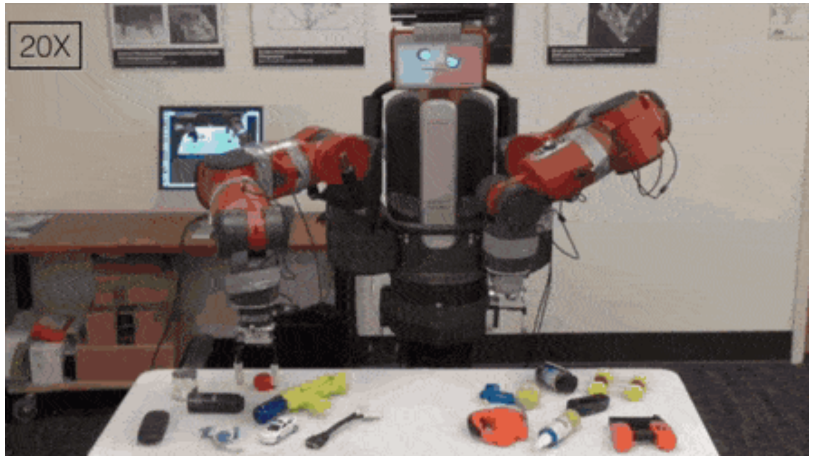
- PÖ genel bir karar verme sistemidir
  - PÖ, **etmenin** bir çevrede **aksiyon** almasıdır
  - Her aksiyon gelecek **durumu** etkiler
  - Etmenin başarısı skaler **ödül** ile ölçülür
  - Amaç: **gelecek ödülleri maksimize eden aksiyonları almak**



PÖ Örnekleri

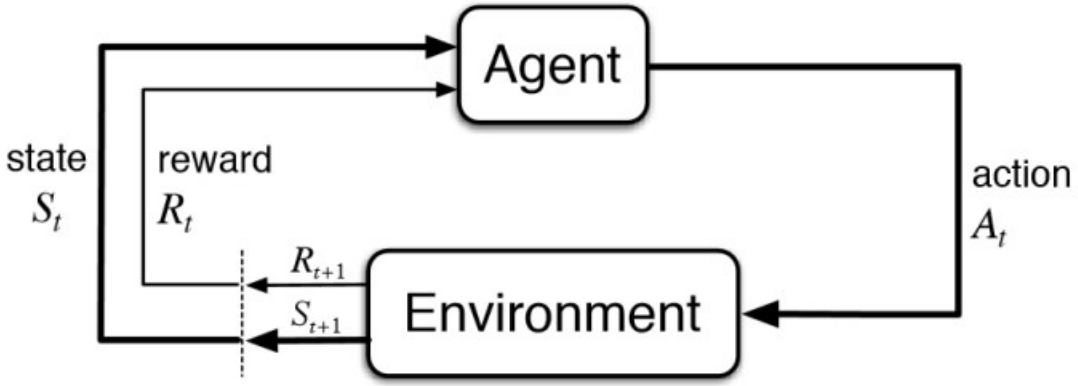
- Oyun oynamak: Atari, Go…
- Kontrol: Manipulasyon, yürümek, uçmak…
- İnsanlarla etkileşim: Öneri, optimizasyon, kişiselleştirme…





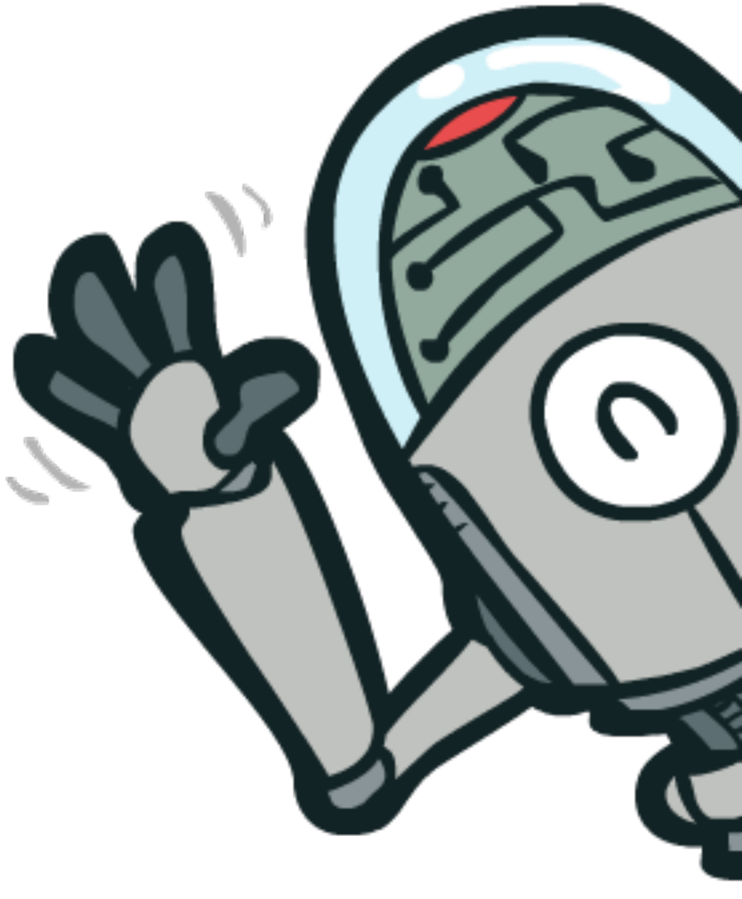
Etmen ve Çevre

- Her zaman noktası *t*'de etmen
  - Aksiyon A_t alır
  - S_t durumunu gözlemler
  - R_t ödülünü elde eder
- Çevre
  - Aksiyon A_t elde eder
  - S_{t+1} durumunu yayınlar
  - R_{t+1} ödülünü yayınlar



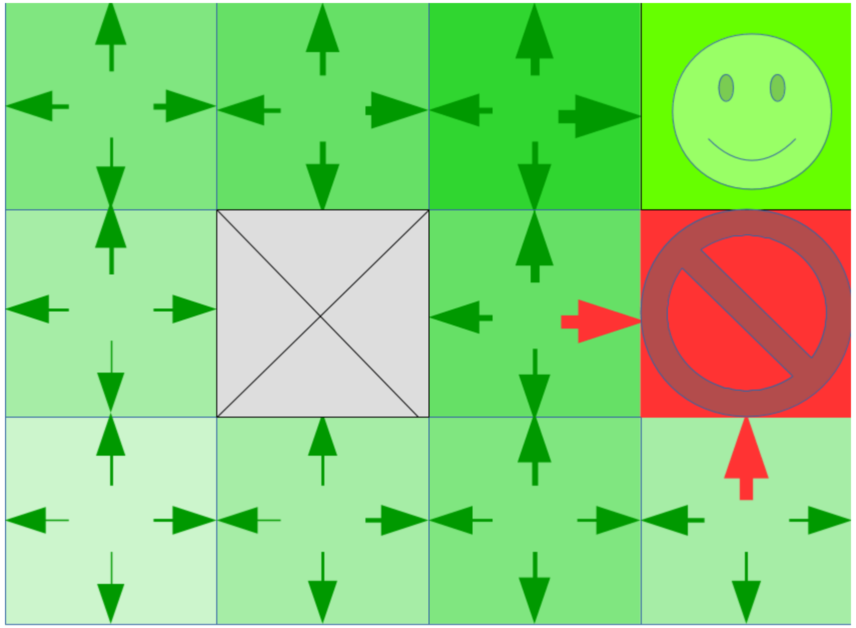
PÖ Etmeni

- Üç temel bileşen
  - **Politika**: Etmenin davranışı
  - **Değer fonksiyonu**: Durum ve/veya aksinonların değerleri
  - **Model**: Etmenin çevreyi temsil etme şekli



Politika

- Etmenin davranışını belirler
- Durumdan aksiyon seçer
  - Deterministik: $a = \pi(s)$
  - Stokastik: $\pi(a|s) = \mathbb{P}[a|s]$



Değer Fonksiyonu

- Değer fonksiyonu, gelecek ödüllerin tahminlenmesidir
  - *"s* durumunda ne kadar ödül kazanırım?"

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

- Aksiyon-değer fonksiyonu
  - *"s* durumunda *a* aksiyonunu alırsam ne kadar ödül kazanırım?"

$$Q^{\pi}(s_t, a_t) = \mathbb{E}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \ldots | s_t, a_t]$$

- Bellman Denklemleri

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}}[r_{t+1} + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) | s_t, a_t]$$

$$V^{\pi}(s_t\ ) = \mathbb{E}_{s_{t+1},}[r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t]$$

Optimal Değer

- Optimal değer fonksiyonu ulaşılabilecek en yüksek değerdir

$$Q^*(s_t, a_t) = \max_{\pi} Q^{\pi}(s_t, a_t)$$

- Optimal değer = optimal aksiyonlar

$$\pi^*(s_t) = arg\max_{a} Q^*(s_t, a)$$

- Bellman Optimallik Denklemi

$$Q^*(s_t, a_t) = \mathbb{E}[r_t + \gamma \max_{a} Q^*(s_{t+1}, a) \,|s_t, a_t]$$

$$V^*(s_t\ ) = \mathbb{E}[r_t + \gamma V(s_{t+1})|s_t, a_t]$$

Model

- Etmenin çevresinin temsili gösterimi
  - Markov varsayımı: Şimdiki durum belli ise gelecek geçmişten bağımsızdır

$$\mathbb{P}[s_{t+1}|s_t] = \mathbb{P}[s_{t+1}|s_t, s_{t-1}, s_{t-2}, \dots, s_1]$$

- Geçiş modeli: $\mathbb{P}[s_{t+1}|s_t, a_t]$
- Ödül modeli: $\mathbb{E}[r_t|s_t, a_t]$
- Genel model: $\mathbb{P}[s_{t+1}, r_t|s_t, a_t]$

Pekiştirmeli Öğrenme Yöntemleri

- Model tabanlı
  - Çevrenin modeli biliniyor
  - Modele bakarak planlama
- Değer tabanlı
  - Optimal değer fonksiyonu bulunur
  - Optimal değer fonksiyonundan politika çıkarılır
- Politika tabanlı
  - Optimal politika bulunur
  - Bu politika maksimum ödülü verir

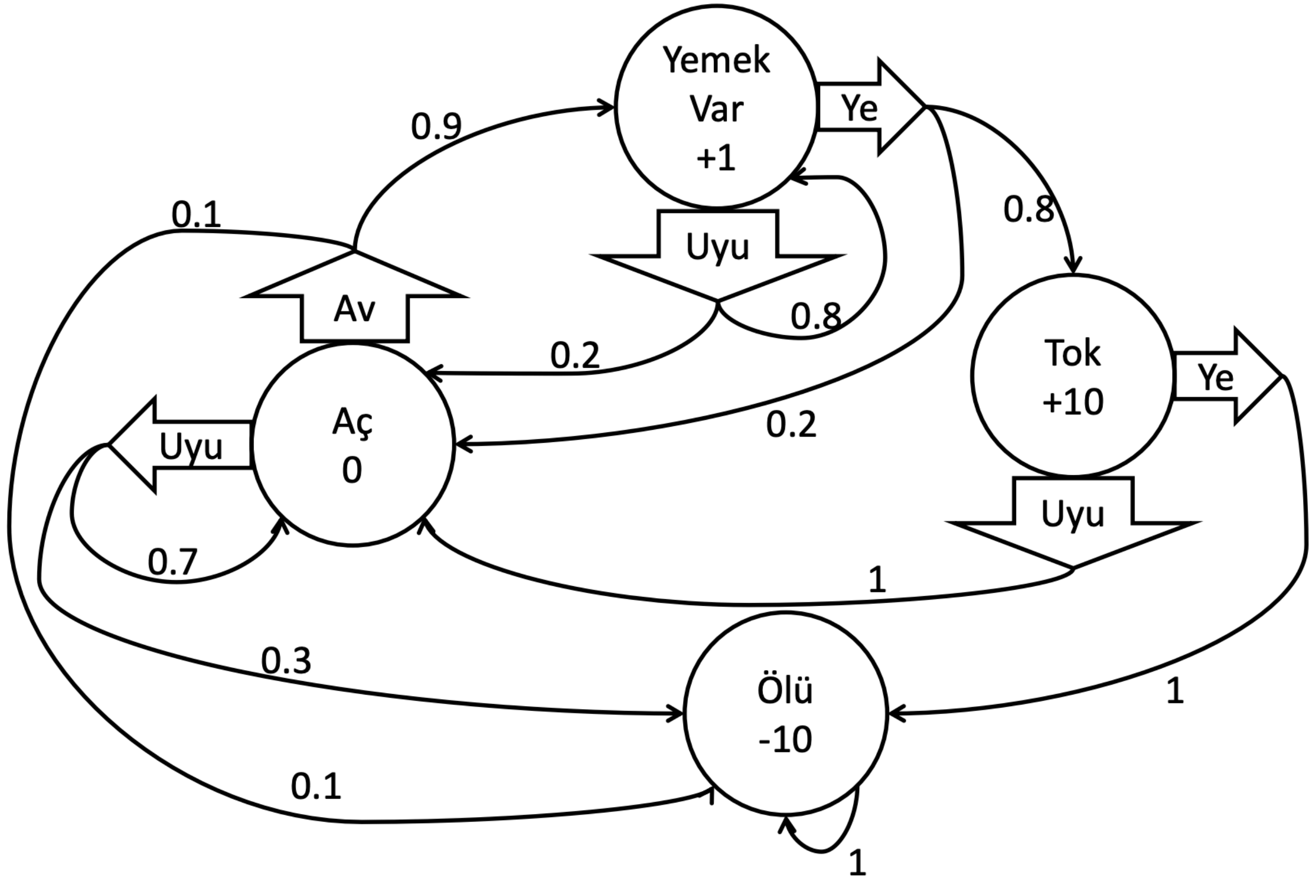
OpenAI Gym

- PÖ araştırma ve geliştirmeleri için geliştirilmiştir
- Etmenlerin yürümekten oyun oynamaya kadar eğitilmesini destekler
  - Ortamlar (Environments)
- Başlamak için
  - https://gym.openai.com/docs/
  - https://github.com/ceteke/kbyoyo_rl/blob/master/OpenAI%20Gym.ipynb
- Ortamlar
  - https://gym.openai.com/envs/

Model Tabanlı

- Markov karar süreci
  - Geçiş fonksiyonu
$$T(s_{t+1}, s_t, a_t) = \mathbb{P}[s_{t+1} | s_t, a_t]$$
  - Ödül fonksiyonu
$$r_t = R(s_t, a_t)$$
  - $\{S, A, R, T, \gamma\}$

Mağara Adamı MKS



MKS Çözümü

- Bellman denkleminin çözümü

$$V^{\pi}(s_t\ ) = \mathbb{E}[r_{t+1} + \gamma V^{\pi}(s_{t+1})|s_t]$$

$$V^{\pi}(s_t) = \sum_{a \in \mathrm{A}} \pi(a|s_t)\,(\,r_{t+1} + \gamma \sum_{s' \in S} \mathbb{P}[s_{t+1}|s_t, a_t]V(s'))$$

MKS Matris Formu

$$V^{\pi} = \boldsymbol{R} + \gamma \boldsymbol{P} V^{\pi}$$

- V her satırında değerleri içeren kolon vektörü
- $\boldsymbol{R}$ her satırında ödülleri içeren kolon vektörü
- $\boldsymbol{P} \in \mathbb{R}^{NxN}$ geçiş olasılıklarını içeren matris

$$V^{\pi} = (I - \gamma \boldsymbol{P})^{-1} \boldsymbol{R}$$

$$O(N^2)$$

- Küçük MKS için kullanışlı

Dinamik Programlama

- Değer İterasyonu (Value Iteratıon)

$$V(s) = 0 \quad \forall_s, \Delta \leftarrow 0$$
$$Tekrarla$$
$$Her\ s \in S$$
$$v \leftarrow V(s)$$
$$V(s) \leftarrow \max_a \sum_{s'} \mathbb{P}[s_{t+1}|s_t, a_t](r_t + \gamma V(s'))$$
$$\Delta \leftarrow \max(\Delta, |v - V(s)|)$$
$$\Delta < \theta' a\ kadar$$

https://github.com/ceteke/kbyoyo_rl/blob/master/Ice%20World%20Value%20Iteration.ipynb

Model

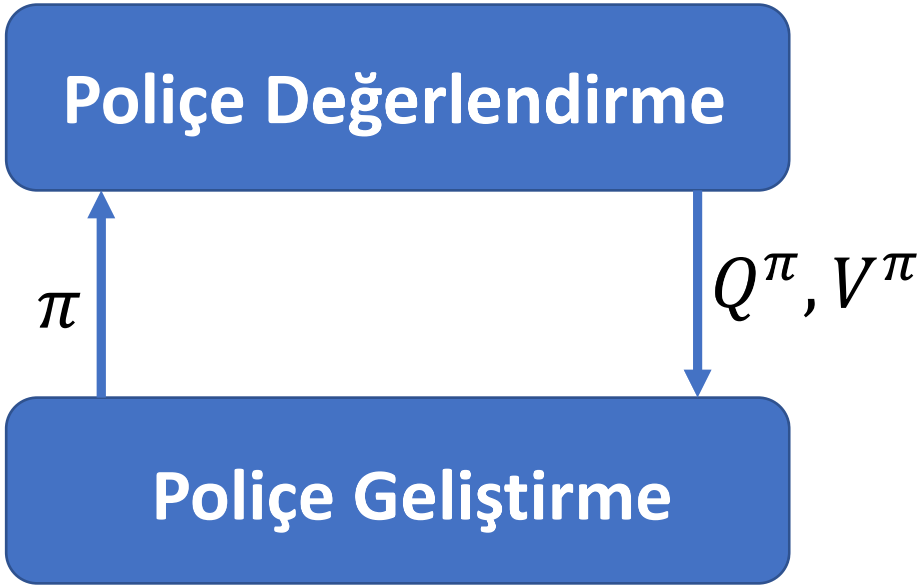
$$\mathbb{P}[s_{t+1}, r_t | s_t, a_t]$$

- Modeli gerçekten bilebilir miyiz?
- Modelimiz ne kadar doğru?

Modelsiz Öğrenme

Modelsiz Öğrenme

- Değer tabanlı öğrenme
- Etmen deneme yanılma ile değerleri öğrenir
- Bir politika ile başlanır, elde edilen değerlere göre politika güncellenir



Zamansal Fark

- Temporal Difference
- Her tecrübeden öğren

$$V^{\pi}(s_t) \leftarrow (1-\alpha)V^{\pi}(s_t) + \alpha\left(r_t + \gamma V^{\pi}(s_{t+1})\right)$$

$$V^{\pi}(s_t) \leftarrow V^{\pi}(s_t) + \alpha\Big(\underbrace{\overbrace{r_t + \gamma V^{\pi}(s_{t+1})}^{\text{Bellman Optimallik}} - V^{\pi}(s_t)}_{\text{Hata}}\Big)$$

$$r_t = R(s_t, \pi(s_t))$$

- Sıradaki durumları bilmeden nasıl aksiyon seçebiliriz?

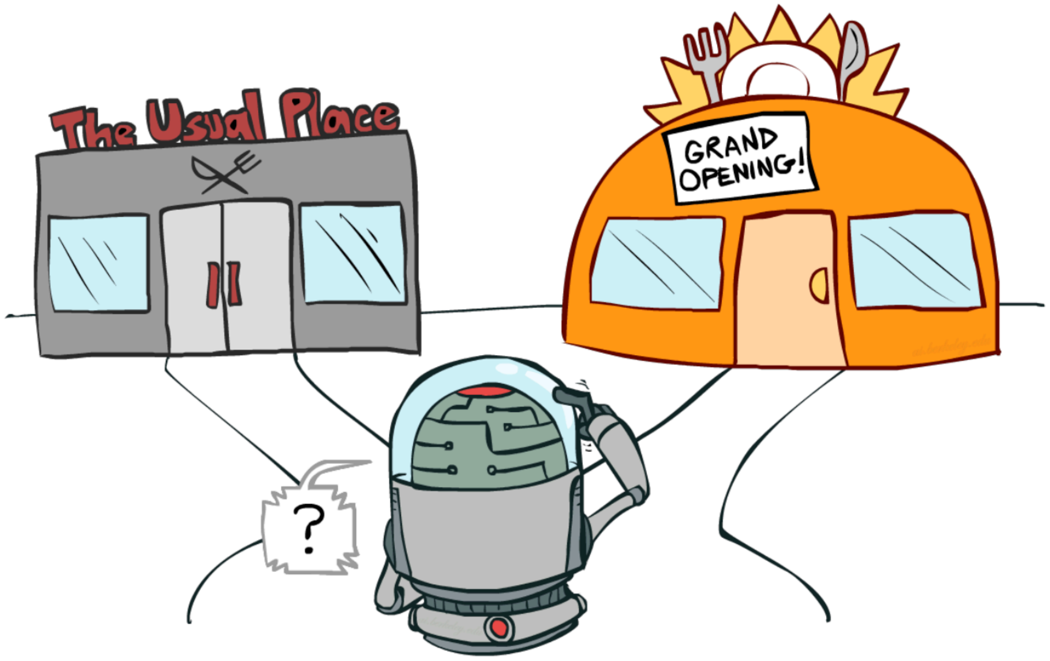
Q-Öğrenmesi

$$Q^{\pi}(s_t, a_t) \leftarrow Q^{\pi}(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q^{\pi}(s_{t+1}, a) - Q^{\pi}(s_t, a_t) \right)$$

- **Politika dışı** öğrenme
- Etmen tecrübeleri nasıl elde edecek?
  - Keşif
- Optimal politika bulma garantisi

Keşif ve Sömürü

- Yeni tecrübeler (keşif)
  - Potensiyel iyi ödüller
- Bilinen yollar (sömürü)
- Hangisi daha iyi?
- ϵ-açgözlü
  - ϵ olasılıkla rasgele hareket seç



SARSA

$$Q^{\pi}(s_t, a_t) \leftarrow Q^{\pi}(s_t, a_t) + \alpha\big(r_t + \gamma\, Q^{\pi}(s_{t+1}, \pi(s_{t+1})) - Q^{\pi}(s_t, a_t)\big)$$

- **Politika içi** öğrenme
- Optimale **yakın** politika bulma garantisi
  - Neden SARSA?

Problemler

- Tablosal methodlar
- Zaman
- Hafıza
- Durumlar aralıksız ise ne yapacağız?
- Genelleme yapılabilir mi?
    - Parametrize etmek

Yaklaşık Öğrenim

- Öznitelik tabanlı
  - Atonom araç öznitelikleri neler olabilir?
- Reel sayılar kullanmak
- Fonksiyon parametreleri öğrenelim

$$V_\theta(s) = w_1 f_1(s) + w_2 f_2(s) + \cdots + w_n f_n(s)$$

$$Q_\theta(s) = w_1 f(s,a) + w_2 f_2(s,a) + \ \ldots + w_n f_n(s,a)$$

- Amaç: Ödülü maksimize eden θ

Yaklaşık Q-Öğrenimi

$$Q^{\pi}(s_t, a_t) \leftarrow Q^{\pi}(s_t, a_t) + \alpha\left(r_t + \gamma \max_a Q^{\pi}(s_{t+1}, a) - Q^{\pi}(s_t, a_t)\right)$$

- Lineer projeksiyon ile aksiyon değerlerinin tahmini

Hata

$$Q_W^{\pi}(s, .) = Ws + b$$

$$\delta = r_t + \gamma \max_a Q_W^{\pi}(s_{t+1}, a) - Q_W^{\pi}(s_t, a_t)$$

- Hatayı nasıl minimize ederiz?
  - Gradyan yönünde ilerleyerek

$$W \leftarrow W + \alpha\delta\nabla_W Q_W(s, a)$$

Ölümcül Üçlü

- Yaklaşık öğrenim
- Poliçe dışı öğrenim
- Önyükleme (bootstrapping)
  - Tahminlenen değeri öğrenim için kullanmak

Yaklaşık SARSA

$$Q^{\pi}(s_t, a_t) \leftarrow Q^{\pi}(s_t, a_t) + \alpha\big(r_t + \gamma\, Q^{\pi}(s_{t+1}, \pi(s_{t+1})) - Q^{\pi}(s_t, a_t)\big)$$

- Politika içi

$$\delta = r_t + \gamma\, Q_W^{\pi}(s_{t+1}, a_{t+1}) - Q_W^{\pi}(s_t, a_t)$$

$$W \leftarrow W + \alpha\delta\nabla_W Q_W(s, a)$$

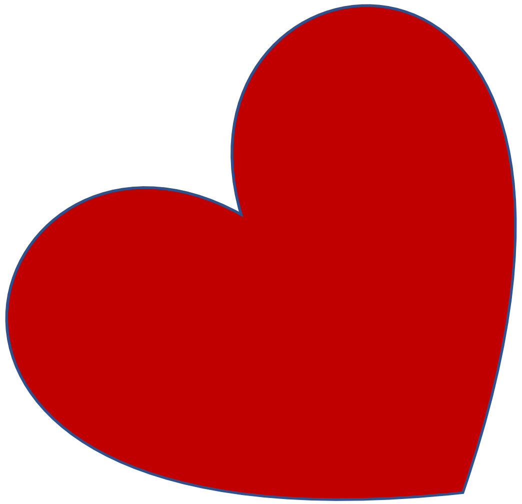
https://github.com/ceteke/kbyoyo_rl/blob/master/Cartpole%20Linear%20Approximation.ipynb

Problemler

- Yüksek boyutlar
- Yakınsamalar “politikayı bozabilir”
- Her adımda korelasyonsuz keşif
  - Tehlikeli
- Aksiyonlar sonsuz sayıda ise ne yapacağız?

Politika Tabanlı Öğrenim

- Politikayı parametrize edelim: $\pi_\theta(a_t|s_t)$
  - Ölçeklenebilir
- Lokal optimal çözümler: $\theta_{yeni} = \theta_{eski} + \alpha \frac{dJ}{d\theta}$
  - Güvenli poliçe
- Güvenli keşif: $\hat{\theta} \sim \mathcal{N}(\theta|\mu_\theta, \Sigma_\theta)$
  - Parametre uzayında



Algoritması

- Üç adım

Tekrarla

1) Keşif: Şu anki politikayı (π_{θ_k}) kullanarak hareket et

2) Değerlendir: Hareketin kalitesini değerlendir

3) Güncelle: Politikayı güncelle ($\pi_{\theta_{k+1}}$)

Yakınsamaya kadar

REINFORCE

- Politika gradyan algoritması
- Amaç fonksiyonu: $J(\theta_k) = \mathbb{E}[\sum_t R(s_t, a_t) | \pi_{\theta_k}]$
- Maksimize etmek içın: $\theta_{k+1} \leftarrow \theta_k + \alpha \nabla_{\theta_k} J(\theta_k)$
- Uzun işlemler sonucu

$$\theta_{k+1} \leftarrow \theta_k + \alpha \nabla_\theta log \pi_\theta (s_t, a_t) G_t$$

$$G_t = \sum_{i=t} \gamma^{i-t} r_{t+i}$$

Lineer Politikalar

- Ayrık aksiyon politikası

$$\pi_\theta(a_t|s_t) = \frac{e^{(Ws)_{a_t}}}{\sum_a e^{(Ws)_a}}$$

- Devamlı aksiyon politikası

$$\mu(s_t) = W_\mu s_t \qquad \sigma(s_t) = e^{W_\sigma s_t}$$

$$\pi_\theta(a_t|s_t) \sim \mathcal{N}(a_t|\mu(s_t), I\sigma(s_t))$$

https://github.com/ceteke/kbyoyo_rl/blob/master/LunarLander%20Policy%20Gradient.ipynb

REINFORCE Problemler

- Varyasyon
- Varyasyon
- Varyasyon
- Her adım için gradyan hesabı
  - Doğal Gradyan (Natural Gradient)

Soru ve Cevap