

A woman with long dark hair is looking down at a tablet computer she is holding. She is wearing a dark patterned shirt. The background is a blurred city street at night, with lights from buildings and streetlights visible through a window or glass partition. The entire image has a blue color overlay.

# ACからDDPG・ SACまで

サブタイトルをここに入力

# 目次

- Actor Critic法
- DDPG法
- Soft Bellman equation
- Soft Q learning
- Soft Actor Critic法

A blue-tinted background image showing several hands pointing at a large sheet of paper, likely a blueprint or architectural plan, suggesting a collaborative work environment.

# ACTOR CRITIC

サブタイトルをここに入力

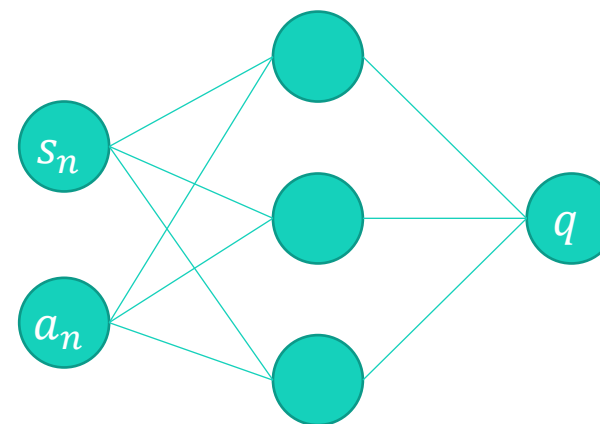
# ACTOR CRITIC

- 状態行動価値関数
  - Q Learningでは状態を受け取り、各行動のQ値を返す

- $f(s_n) = (q_{a_1}, q_{a_2}, \dots, q_{a_m})$
- 行動は離散的である必要あり

	$a_1$	$a_2$	$a_3$
$s_1$	10	20	30
$s_2$	15	10	25
$s_3$	20	15	5

- Actor Criticでは状態と行動を受け取り、Q値を返す
  - $g(s_n, a_m) = q$
  - この関数 $g$ は状態と行動の価値を判定しているためCriticと呼ぶ
  - 連続的な行動も許容



# ACTOR CRITC

---

- 方策関数 $\pi$ 
  - Q Learningでは状態に対する各行動のQ値を基に $\epsilon$ -greedy法で探索

$$f(s_n) = (q_{a_1}, q_{a_2}, \dots, q_{a_m})$$
$$a = \begin{cases} \operatorname{argmax}(q_{a_1}, q_{a_2}, \dots, q_{a_m}) & \text{for } 1 - \epsilon \\ \operatorname{random}(a_1, a_2, \dots, a_m) & \text{for } \epsilon \end{cases}$$

- Actor Criticでは方策関数を直接モデル化
  - この方策関数をactorと呼ぶ
  - 方策関数が確定的な関数の場合、探索のためランダム項を加えることが多い

$$a = \pi(s_n)$$

# POINTS

- Actor Critic法は連続的な行動を扱うことができる
- Criticとは状態と行動を受け取り状態行動価値を返す
- Actorとは状態を受け取り行動を返す





# DDPG

サブタイトルをここに入力

# DDPG

---

- ここではActor Critic法の実装の一つであるDeep Deterministic Policy Gradient法について紹介する
  - ここでは連続的な行動を仮定したケースについて考える
- DDPG法では状態行動価値関数・方策関数をそれぞれDeep Neural Networkで表現する
  - 状態行動価値関数:  $q = Q(s, a)$
  - 方策関数:  $a = \pi(s)$
- 探索方法
  - DDPGでは方策関数を決定的な関数として実装している
  - そのため、探索時はランダム性を考慮するためにノイズ項を入れる
    - $a_t^{exploration} = \pi(s_t) + \varepsilon$
  - 探索結果である $(s_t, a_t, r_t, s_{t+1})$ をメモリに保存する点はDQNと同様



# DDPG

---

- 最適化方法
  - Criticの更新はDQNと同様にBellman誤差を減少させるように最適化
    - Bellman誤差:  $Q(s_t, a_t) - \{r_t + \gamma Q(s_{t+1}, a_{t+1}^{exploitation})\}$
    - $a_{t+1}^{exploitation}$  はノイズ項なしの方策関数から求める  $a_{t+1}^{exploitation} = \pi(s_{t+1})$
    - 探索と活用で用いる方策が異なることから本手法はoff-policy
  - Actorの更新時はQ値を最大化させるように方策関数を最適化
    - Q値:  $Q(s_t, \pi(s_t)) = Q(s_t, a_t^{exploitation})$
    - 実装時はマイナスQ値をlossとして最小化させるように最適化

# DDPG


---

- 実装を試みる

# POINTS

- DDPG法はoff-policyな手法
- CriticとActorをDNNで表現
- 連続的な行動も取り扱い可能





# SOFT BELLMAN EQUATION

サブタイトルをここに入力

# SOFT BELLMAN EQUATION

- エントロピーの定義

- 本節では確率の方策 $\pi(a|s)$ について考える。 $\pi(a|s)$ は状態 $s$ での行動の確率密度関数である
- このとき方策 $\pi(a|s)$ に対するエントロピーは次のように定義される
  - 離散エントロピー:  $H(\pi|s) = -\sum_a \pi(a|s) \ln \pi(a|s)$
  - 連続エントロピー:  $H(\pi|s) = -\int_{-\infty}^{\infty} \pi(a|s) \ln \pi(a|s) da = -E[\ln \pi(a|s)]$

- 例1: 右に進む確率を $x$ 、左に進む確率を $1 - x$ とする。このときエントロピーは次の通り

$$H(\pi|s) = -x \ln x - (1 - x) \ln(1 - x)$$

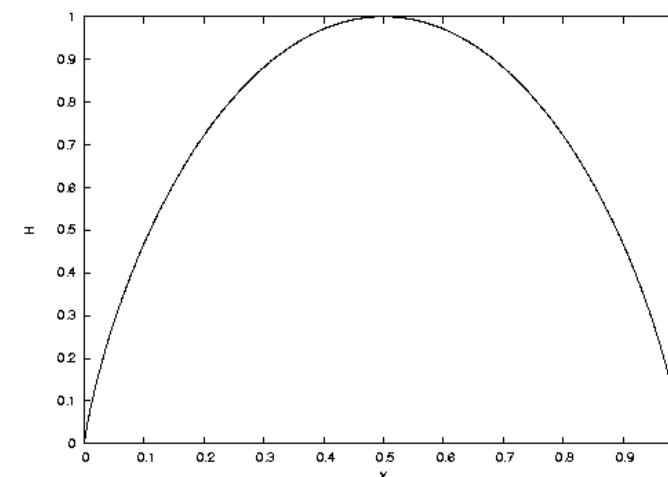
- 例2:  $\pi(a|s)$ が平均 $\mu$ 、分散 $\sigma^2$ の正規分布とする。このときエントロピーは次の通り

$$H(\pi|s) = \ln(\sigma\sqrt{2\pi e})$$

- エントロピーの性質

- エントロピーは確率分布の平均に依存せず分散にのみ依存
- 分散が増大するとエントロピーも増大

例1のグラフ



# SOFT BELLMAN EQUATION

---

- 一般のMDPは割引期待報酬和を最大化する方策を見つけることが目的

$$\max_{\pi} E \left[ \sum_{i=0}^{\infty} \gamma^i r(s_{t+i}, a_{t+i}) | s_t \right]$$

- Soft MDPでは報酬+エントロピーを最大化する方策を見つける
  - $\alpha$ はエントロピー正則化の定数

$$\max_{\pi} E \left[ \sum_{i=0}^{\infty} \gamma^i (r(s_{t+i}, a_{t+i}) + \alpha H(\pi | s_{t+i})) | s_t \right]$$

- エントロピーは方策のばらつきや不安定性を表しており、局所解に陥らないように探索することができる
  - →ロバスト性の向上

# SOFT BELLMAN EQUATION

---

- Soft value function

$$\begin{aligned} V(s_t) &= E \left[ \sum_{i=0}^{\infty} \gamma^i (r(s_{t+i}, a_{t+i}) + \alpha H(\pi|s_{t+i})) | s_t \right] \\ &= E \left[ r(s_t, a_t) + \alpha H(\pi|s_t) + \gamma \sum_{i=0}^{\infty} \gamma^i (r(s_{t+1+i}, a_{t+1+i}) + \alpha H(\pi|s_{t+1+i})) | s_t \right] \\ &= E[r(s_t, a_t) + \alpha H(\pi|s_t) + \gamma V(s_{t+1}) | s_t] \end{aligned}$$

- Soft Q function

$$\begin{aligned} Q(s_t, a_t) &= E \left[ \sum_{i=0}^{\infty} \gamma^i (r(s_{t+i}, a_{t+i}) + \alpha H(\pi|s_{t+i})) | s_t, a_t \right] \\ &= E[r(s_t, a_t) + \gamma V(s_{t+1}) | s_t, a_t] \\ &= E[r(s_t, a_t) + \gamma \{Q(s_{t+1}, a_{t+1}) + \alpha H(\pi|s_{t+1})\} | s_t, a_t] \end{aligned}$$

- Q and value function

$$V(s_t) = E[Q(s_t, a_t) + \alpha H(\pi|s_t) | s_t]$$

# POLICY ITERATION

---

- 方策関数をsoft q functionのsoftmax関数として定義する

$$\pi(a_t|s_t) = \frac{\exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right)}{\int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t}$$

- 方策関数をvalue functionの式に代入

$$\begin{aligned} V(s_t) &= E \left[ Q(s_t, a_t) - \alpha \ln \left( \frac{\exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right)}{\int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t} \right) | s_t \right] \\ &= E \left[ \alpha \ln \int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t | s_t \right] \end{aligned}$$

- この期待値は行動 $a_t$ について独立なため

$$V(s_t) = \alpha \ln \int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t$$



# POLICY ITERATION

---

- Q値とvalueをそれぞれ次のように収束するまで更新

$$Q(s_t, a_t) \leftarrow E[r(s_t, a_t) + \gamma V(s_{t+1}) | s_t, a_t]$$
$$V(s_t) \leftarrow \alpha \ln \int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t$$

- 最適な方策は最適なQ値 $Q^*$ とvalue  $V^*$ を用いて次のように求める

$$\pi^*(a_t | s_t) = \frac{\exp\left(\frac{1}{\alpha} Q^*(s_t, a_t)\right)}{\int \exp\left(\frac{1}{\alpha} Q^*(s_t, a_t)\right) da_t}$$
$$= \exp\left(\frac{1}{\alpha} (Q^*(s_t, a_t) - V^*(s_t))\right)$$

- しかし、この更新は二つの理由で上手くいかない
  - Soft Bellman 更新は連続または大きな状態・行動空間で正確に最適化されない
  - 最適なQ値とvalueから最適方策を計算することがtractableでない

# POINTS

- エントロピー正則化項を導入することで、価値関数を最適化すると方策関数も最適化される
- Soft Bellman equationの枠組みでは方策をsoftmax関数で表す
- Soft Bellman equationは工夫なしではうまくワークしない





# SOFT Q LEARNING

サブタイトルをここに入力

# SOFT Q LEARNING

---

- Valueの更新式はintractable

$$V(s_t) \leftarrow \alpha \ln \int \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right) da_t$$

- しかしながら、importance samplingを用いることで次のように更新可能
  - $q(\mathbf{a})$ は行動 $\mathbf{a}$ をとる確率

$$V(s_t) = \alpha \ln E_{\mathbf{a}} \left[ \frac{\exp\left(\frac{1}{\alpha} Q(s_t, \mathbf{a})\right)}{q(\mathbf{a})} \right]$$

- 行動価値関数をパラメータ $\theta$ でモデル化 $Q_{\theta}(s_t, a_t)$ とすると、次のBellman誤差を最小化するように最適なQ値を求める

$$Loss\{Q_{\theta}(s_t, a_t), r(s_t, a_t) + \gamma V(s_{t+1})\}$$

# SOFT Q LEARNING

---

- 方策関数は次のKL divergenceを最小化するように最適化を行う

$$\begin{aligned} & D_{KL} \left( \pi^\theta(a_t|s_t) \parallel \exp \left( \frac{1}{\alpha} (Q(s_t, a_t) - V(s_t)) \right) \right) \\ &= D_{KL} \left( \pi^\theta(a_t|s_t) \parallel \exp \left( \frac{1}{\alpha} Q(s_t, a_t) \right) \right) \end{aligned}$$

# POINTS

- 価値関数はsoft Bellman誤差を最適化する
- 方策関数の最適化はKL divergenceを用いて定式化される





# SOFT ACTOR CRITIC

サブタイトルをここに入力

# SOFT ACTOR CRITIC

---

- 価値関数の最適化はSOFT Q functionの式を用いて行う

$$Q(s_t, a_t) = E[r(s_t, a_t) + \gamma\{Q(s_{t+1}, a_{t+1}) + \alpha H(\pi|s_{t+1})\}|s_t, a_t]$$

より

$$Loss = \{Q(s_t, a_t), r(s_t, a_t) + \gamma\{Q(s_{t+1}, a_{t+1}) - \alpha \ln \pi(s_{t+1})\}\}$$

- 方策関数はKL divergenceから最適化を行う

$$D_{KL}\left(\pi^\theta(a_t|s_t) \parallel \exp\left(\frac{1}{\alpha} Q(s_t, a_t)\right)\right)$$

より

$$Loss = \{Q(s_t, a_t), \alpha \ln \pi^\theta(a_t|s_t)\}$$



# REPARAMETERIZATION TRICK

---

- Soft Actor Criticでは確定的方策を用いるも、reparameterization trickを用いて疑似的に確率的な方策を用いる
  - 確定的な方策関数 $\pi(a_t|s_t)$ に対し、行動をノイズ項 $\epsilon_t$ を引数とした関数 $f(\epsilon_t; s_t)$ で近似
$$\pi(f(\epsilon_t; s_t)|s_t)$$
  - $\epsilon_t$ に確率変数を入力することで確率的な方策となり、定数を入力すると確定的な方策となる

- 具体的には、平均 $\mu$ 、分散 $\sigma^2$ に対し、行動を以下のように近似することが一般的
  - このtanh関数をsquashing functionと呼ぶ
  - ここで、 $\epsilon_t$ は標準正規乱数であり、 $\mu_t$ と $\sigma_t$ は $s_t$ に依存して決まる
$$a_t = \tanh(\mu_t + \sigma_t \epsilon_t)$$

- つまり、平均 $\mu$ 、分散 $\sigma^2$ の正規乱数 $x_t$ に対し、 $a_t = \tanh(x_t)$ である
  - このときの $a_t$ の確率密度関数 $\pi(a_t|s_t)$ はどんな形になるだろうか？

$$\begin{aligned}\pi(a_t|s_t) &= \left( \frac{\partial \tanh x_t}{\partial x_t} \right)^{-1} \cdot \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(-\frac{(x_t - \mu_t)^2}{2\sigma_t^2}\right) \\ &= \frac{1}{1 - \tanh^2 x_t} \cdot \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(-\frac{(x_t - \mu_t)^2}{2\sigma_t^2}\right)\end{aligned}$$

# REPARAMETERIZATION TRICK

---

- したがって、価値関数と方策関数の更新式に現れる $\ln \pi$ は以下のように計算される

$$\begin{aligned}\ln \pi(a_t|s_t) &= \ln \left\{ \frac{1}{1 - \tanh^2 x_t} \cdot \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp \left( -\frac{(x_t - \mu_t)^2}{2\sigma_t^2} \right) \right\} \\ &= \ln \left( \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp \left( -\frac{(x_t - \mu_t)^2}{2\sigma_t^2} \right) \right) - \ln(1 - \tanh^2 x_t) \\ &= \ln \left( \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp \left( -\frac{(x_t - \mu_t)^2}{2\sigma_t^2} \right) \right) - \ln \left( \frac{4}{e^{2x_t} + e^{-2x_t} + 2} \right) \\ &= \ln \left( \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp \left( -\frac{(x_t - \mu_t)^2}{2\sigma_t^2} \right) \right) - \{2 \ln 2 - 2 \ln(e^{x_t} + e^{-x_t})\} \\ &= \ln \left( \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp \left( -\frac{(x_t - \mu_t)^2}{2\sigma_t^2} \right) \right) - 2\{\ln 2 - x_t - \ln(1 + e^{-2x_t})\}\end{aligned}$$

- コードを見返してみる

# POINTS

- Soft actor criticではQ値と方策関数をモデル化
- 方策関数の最適化はParameterization trickという手法を用いて行う







# APPENDIX

強化学習におけるエントロピー正則化以外の正則化法





# 強化学習における正則化法

- （強化学習を問わず）主な正則化法
  - L1, L2正則化法
  - Batch normalization
  - Drop out
- 強化学習において、上記の正則化法はBellman誤差の最適化の正則化よりはDNNモデルの正則化手法として用いられる
- エントロピー正則化以外の正則化法をBellman方程式に組み込むことは未だされていない模様

# REFERENCE



## OPEN AI

<https://spinningup.openai.com/en/latest/algorithms/sac.html#>



## SOFT ACTOR-CRITIC: OFF-POLICY MAXIMUM ENTROPY DEEP REINFORCEMENT LEARNING WITH A STOCHASTIC ACTOR

<https://arxiv.org/pdf/1801.01290.pdf>



## DDPG CODE

<https://github.com/openai/spinningup/tree/master/spinup/algos/pytorch/ddpg>



## REINFORCEMENT LEARNING WITH DEEP ENERGY-BASED POLICIES

<https://arxiv.org/pdf/1702.08165.pdf>



## MAXIMUM ENTROPY REINFORCEMENT LEARNING (STOCHASTIC CONTROL)

<https://www.slideshare.net/DongMinLee32/maximum-entropy-reinforcement-learning-stochastic-control>



## SAC CODE

<https://github.com/openai/spinningup/tree/master/spinup/algos/pytorch/sac>