به نام خدا

دانشگاه تهران

پردیس دانشکده های فنی

دانشکده مهندسی برق و کامپیوتر

تمرین شماره‌ی 6

بخش کتبی

مدرسین:

دکتر فدائی

دکتر یعقوب‌زاده

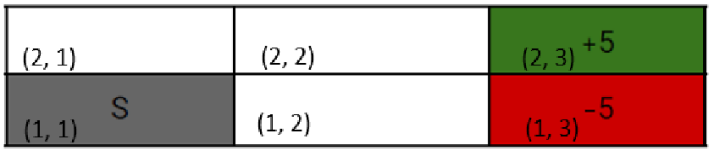
نگارش:

فاطمه محمدی 810199489

بخش کتبی

MDP

سوال اول



Transaction Function: $P(S|A,S') = 0.8 \rightarrow T(S,A,S') = 0.8$

(1

Discount Factor: $\gamma = 0.9$ $V_{i+1}(s) = \max_a(\sum_s T(s,a,s')(R(s,a,s') + \gamma V_i(s')))$

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
V_0	0	0	5	0	0	-5
V_1			5			-5
V_2			5			-5

V_1:

$V_1(1,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0
→	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0
↓	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_1(1,1) = max(0,0,0,0) = 0}$$

$V_1(2,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0
→	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0
↓	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_1(2,1) = max(0,0,0,0) = 0}$$

$V_1(1,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*-5) =-0.45	-	0.1*(0 + 0.9*0) = 0	-0.45
→	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*-5) =-0.36	0.1*(0 + 0.9*0) = 0	-	-3.6
↓	-	0.1*(0 + 0.9*-5) =-0.45	0.1*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-0.45
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_1(1,2) = \max(-0.45, -3.6, -0.45, 0) = 0}$$

$V_1(2,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*5) = 0.45	-	0.1*(0 + 0.9*0) = 0	0.45
→		0.8*(0 + 0.9*5) = 3.6	0.1*(0 + 0.9*0) = 0	-	3.6
↓	-	0.1*(0 + 0.9*5) = 0.45	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0.45
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_1(2,2) = \max(0.45, 3.6, 0.45, 0) = 3.6}$$

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
$\mathbf{V_0}$	0	0	5	0	0	-5
$\mathbf{V_1}$	0	0	5	0	3.6	-5
$\boldsymbol{V_2}$			5			-5

V_2:

$V_2(1,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0
→	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-	0
↓	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_2(1,1) = \max(0, 0, 0, 0) = 0}$$

$V_2(2,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*3.6) = 0.324	-	0.1*(0 + 0.9*0) = 0	0.324
→	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*3.6) = 2.592	0.1*(0 + 0.9*0) = 0	-	2.592
↓	-	0.1*(0 + 0.9*3.6) = 0.324	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0.324
←	0.1*(0 + 0.9*0) = 0	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0

$$\mathbf{V_2(2,1) = max(0.324, 2.592, 0.324, 0) = 2.592}$$

$V_2(1,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*3.6) = 2.592	0.1*(0 + 0.9*-5) = -0.45	-	0.1*(0 + 0.9*0) = 0	2.142
→	0.1*(0 + 0.9*3.6) = 0.324	0.8*(0 + 0.9*-5) = -3.6	0.1*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-3.276
↓	-	0.1*(0 + 0.9*-5) = -0.45	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	-0.45
←	0.1*(0 + 0.9*3.6) = 0.324	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.324

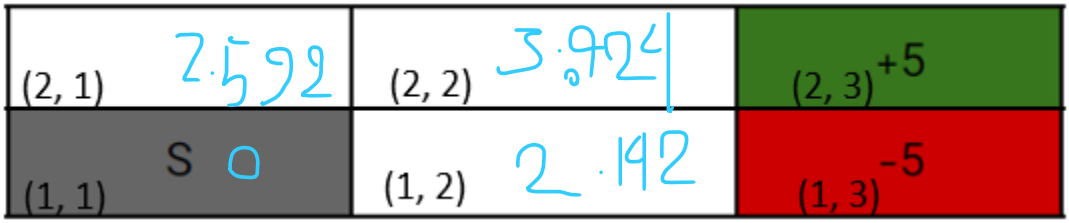
$$\mathbf{V_2(1,2) = max(2.142, -3.276, -0.45, 0.324) = 2.142}$$

$V_2(2,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*3.6) = 2.592	0.1*(0 + 0.9*-5) = 0.45	-	0.1*(0 + 0.9*0) = 0	3.042
→	0.1*(0 + 0.9*3.6) = 0.324	0.8*(0 + 0.9*-5) = 3.6	0.1*(0 + 0.9*0) = 0	-	3.924
↓	-	0.1*(0 + 0.9*-5) = 0.45	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0.45
←	0.1*(0 + 0.9*3.6) = 0.324	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.324

$$\mathbf{V_2(2,2) = max(3.042, 3.924, 0.45, 0.324) = 3.924}$$

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
$\mathbf{V_0}$	0	0	5	0	0	-5
$\mathbf{V_1}$	0	0	5	0	3.6	-5
$\mathbf{V_2}$	0	2.142	5	2.592	3.924	-5



(2

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
$\boldsymbol{\pi^*(s)}$	↑	↑	-	→	→	-

$\mathbf{V_3}$:

$V_3(1,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*2.592)	0.1*(0 + 0.9*2.142)	-	0.1*(0 + 0.9*0) = 0	2.05902
→	0.1*(0 + 0.9*2.592)	0.8*(0 + 0.9*2.142)	0.1*(0 + 0.9*0) = 0	-	1.90734
↓	-	0.1*(0 + 0.9*2.142)	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*0) = 0	0.19278
←	0.1*(0 + 0.9*2.592)	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*0) = 0	0.19278

$V_2(2,1) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*2.592)	0.1*(0 + 0.9*3.924)	-	0.1*(0 + 0.9*2.592)	2.45268
→	0.1*(0 + 0.9*2.592)	0.8*(0 + 0.9*3.924)	0.1*(0 + 0.9*0) = 0	-	3.05856
↓	-	0.1*(0 + 0.9*3.924)	0.8*(0 + 0.9*0) = 0	0.1*(0 + 0.9*2.592)	0.58644
←	0.1*(0 + 0.9*2.592)	-	0.1*(0 + 0.9*0) = 0	0.8*(0 + 0.9*2.592)	2.09952

$V_2(1,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*3.924)	0.1*(0 + 0.9*-5)	-	0.1*(0 + 0.9*0) = 0	2.37528

→	0.1*(0 + 0.9*3.924)	0.8*(0 + 0.9*-5)	0.1*(0 + 0.9*2.142)	-	-3.05406
↓	-	0.1*(0 + 0.9*-5)	0.8*(0 + 0.9*2.142)	0.1*(0 + 0.9*0) = 0	1.09224
←	0.1*(0 + 0.9*3.924)	-	0.1*(0 + 0.9*2.142)	0.8*(0 + 0.9*0) = 0	0.54594

$V_2(2,2) =$

	↑	→	↓	←	SUM
↑	0.8*(0 + 0.9*3.924)	0.1*(0 + 0.9*5)	-	0.1*(0 + 0.9*2.592)	3.50856
→	0.1*(0 + 0.9*3.924)	0.8*(0 + 0.9*5)	0.1*(0 + 0.9*2.142)	-	4.14594
↓	-	0.1*(0 + 0.9*5)	0.8*(0 + 0.9*2.142)	0.1*(0 + 0.9*2.592)	2.22552
←	0.1*(0 + 0.9*3.924)	-	0.1*(0 + 0.9*2.142)	0.8*(0 + 0.9*2.592)	2.41218

(3

Ep1:

$$(1,1) \rightarrow (1,2) \rightarrow (1,3)$$

Ep2:

$$(1,1) \rightarrow (1,2) \rightarrow (2,2) \rightarrow (2,3)$$

Ep3:

$$(1,1) \rightarrow (2,1) \rightarrow (2,2) \rightarrow (2,3)$$

جدول اولیه:

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
V_0	0	0	5	0	0	-5

براساس جدول اولیه داریم:

$$V(1,1): \frac{1}{3}\,(0 - 5 + 0 + 0 + 5 + 0 + 0 + 5) = 1.66$$

$$V(2,2): \frac{1}{2}\,(5 + 5) = 5$$

---

4) امتیازی

$$V_\pi(s) \leftarrow V_\pi(s) + \alpha(sample - V_\pi(s))$$

1) $V_\pi(1,1) = 0 + 0.1(0 + 0 * 0.9 - 0) = 0$ (EP1)

$V_\pi(1,2) = 0 + 0.1(0 + 0.9 * (-5) - 0) = -0.45$ (EP1)

2) $V_\pi(1,1) = 0 + 0.1(0 + 0.9 * (-0.45) - 0) = -0.0405$ (EP2)

$V_\pi(1,2) = -0.45 + 0.1(0 + 0.9 * (0) - (-0.45)) = -0.405$ (EP2)

$V_\pi(2,2) = 0 + 0.1(0 + 0.9 * (5) - (0)) = -0.45$ (EP2)

S	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)
V_0	0	0	5	0	0	-5
V_1	0	-0.45	5	0	0	-5
V_2	-0.0405	-0.405	5	0	0.45	-5

DQN

سوال اول

مروری بر DQN

الگوریتم Deep Q-Network (DQN) توسط تیم DeepMind توسعه یافته و ترکیبی از الگوریتم یادگیری تقویتی Q-learning با شبکه‌های عصبی عمیق است. این الگوریتم در مقاله‌ای مهم توسط مینخ و همکارانش در سال 2015 معرفی شد. در ادامه به توضیح جامع DQN و کاربردهای آن می‌پردازیم:

بازبینی Q-Learning

Q-learning یک الگوریتم یادگیری تقویتی model-free است که سعی در یادگیری ارزش جفت‌های state-action دارد که با تابع $Q(s, a)$ نشان داده می‌شود. هدف آن یادگیری سیاستی (policy) است که پاداش تجمعی را با دنبال کردن سیاست بهینه‌ای که از تابع Q مشتق شده، به حداکثر برساند.

اجزای کلیدی DQN

1. شبکه عصبی: به جای استفاده از جدولی برای ذخیره مقادیر Q، DQN از یک شبکه عصبی برای تقریب تابع Q استفاده می‌کند. ورودی این شبکه حالت(state) است و خروجی مقادیر Q برای تمامی اعمال(action ها)ی ممکن است.

2. تجربه بازپخش (Experience Replay): DQN تجربیات agent (حالت، عمل، پاداش، حالت بعدی) را در یک حافظه ذخیره می‌کند. در طول آموزش، دسته‌های تصادفی از این حافظه نمونه‌گیری می‌شوند تا همبستگی بین تجربیات متوالی را بشکنند که این کار به تثبیت آموزش کمک می‌کند.

3. شبکه هدف (Target Network): برای جلوگیری از نوسانات و انحراف در مقادیر Q، DQN از یک شبکه هدف جداگانه برای تولید مقادیر هدف Q استفاده می‌کند. وزن‌های شبکه هدف به طور دوره‌ای به روز می‌شوند تا با وزن‌های شبکه اصلی مطابقت داشته باشند.

مراحل الگوریتم DQN

محیط: DQN با محیطی تعامل می‌کند که شامل یک حالت، فضای عمل و تابع پاداش است. هدف DQN یادگیری سیاست بهینه است که پاداش‌های تجمعی را در طول زمان به حداکثر برساند.

Replay Memory: DQN از یک بافر حافظه بازپخش برای ذخیره تجربیات گذشته استفاده می‌کند. هر تجربه یک چهارگانه (حالت، عمل، پاداش، حالت بعدی) است که نشان‌دهنده انتقال از یک حالت به حالت دیگر است. این حافظه تجربیات را ذخیره می‌کند تا بعدها به‌صورت تصادفی نمونه‌گیری شود.

شبکه عصبی عمیق: DQN از یک شبکه عصبی عمیق برای تخمین مقادیر Q برای هر جفت (حالت، عمل) استفاده می‌کند. شبکه عصبی حالت را به‌عنوان ورودی می‌گیرد و مقدار Q برای هر عمل را به‌عنوان خروجی ارائه می‌دهد. شبکه برای به حداقل رساندن تفاوت بین مقادیر Q پیش‌بینی‌شده و مقادیر هدف آموزش داده می‌شود.

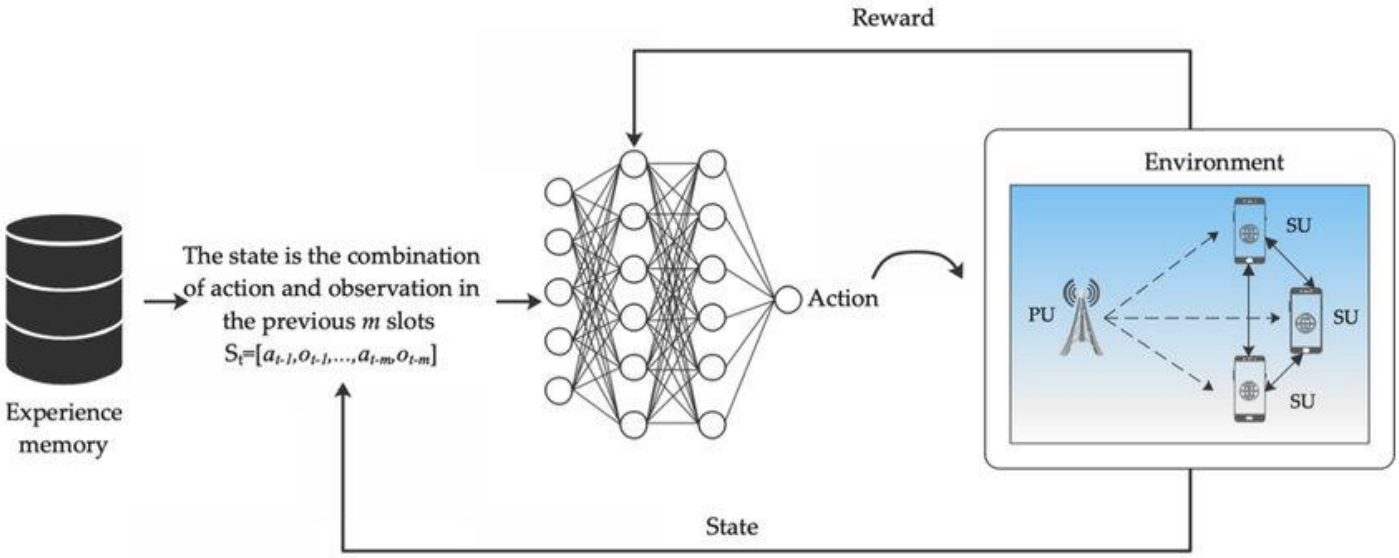
اکتشاف اپسیلون-حریص: DQN از یک استراتژی اکتشاف اپسیلون-حریص برای تعادل بین اکتشاف و بهره‌برداری استفاده می‌کند. در طول آموزش، عامل با احتمال اپسیلون یک عمل تصادفی انتخاب می‌کند و با احتمال (1 - اپسیلون) عمل با بالاترین مقدار Q را انتخاب می‌کند.

شبکه هدف: DQN از یک شبکه هدف جداگانه برای تخمین مقادیر Q هدف استفاده می‌کند. شبکه هدف یک نسخه کپی از شبکه عصبی اصلی با پارامترهای ثابت است. شبکه هدف به‌طور دوره‌ای به‌روزرسانی می‌شود تا از برآورد بیش از حد مقادیر Q جلوگیری کند.

آموزش: DQN شبکه عصبی را با استفاده از معادله Bellman برای تخمین مقادیر Q بهینه آموزش می‌دهد. تابع از دست دادن میانگین مربعات خطا بین مقادیر پیش‌بینی‌شده و مقادیر هدف است. مقدار Q هدف با استفاده از شبکه هدف و معادله Bellman محاسبه می‌شود. وزن‌های شبکه عصبی با استفاده از پس‌انتشار و گرادیان نزولی تصادفی به‌روزرسانی می‌شوند.

آزمایش: DQN بعد از آموزش از سیاست یادگرفته‌شده برای تصمیم‌گیری در محیط استفاده می‌کند. عامل برای یک حالت خاص عمل با بالاترین مقدار Q را انتخاب می‌کند.

(منبع برای مطالعه دقیق تر این قسمت: لینک)



مقایسه با **Q-learning**:

	Q-learning	Deep Q-learning	Deep Q-network
Approach	Tabular learning using Q-table	Function approximation with neural networks	Function approximation with neural networks
Input	(state, action) pairs	Raw State input	Raw State input
Output	Q-values for each (state, action) pair	Q-values for each (state, action) pair	Q-values for each (state, action) pair
Training data	Q-table entries	Experience Replay buffer	Experience Replay buffer
Training time	Fast	Slow	Slow
Complexity	Limited by the number of states and actions	More complex due to the use of neural networks	More complex due to the use of neural networks
Generalization	Limited to states in Q-table	Can generalize to unseen states	Can generalize to unseen states
Scalability	Struggles with large state and action spaces	Handles large spaces well	Handles large spaces well
Stability	Prone to overfitting	More stable than Q-learning, but can still be unstable	More stable than Q-learning and deep Q-learning

کاربردهای **DQN**

. بازی‌ها: DQN به طور معروف برای دستیابی به عملکرد در سطح انسانی در بازی‌های Atari 2600 استفاده شد، که سیاست‌ها را مستقیماً از ورودی‌های پیکسلی خام یاد می‌گیرد.

2. رباتیک: DQN می‌تواند برای وظایف کنترل روباتیک استفاده شود، جایی که agent یاد می‌گیرد در محیط فیزیکی اعمال را انجام دهد.

3. مالی: DQN می‌تواند در استراتژی‌های معاملاتی استفاده شود، جایی که حالت می‌تواند شرایط بازار را نمایندگی کند و اعمال می‌توانند تصمیمات خرید، فروش یا نگه‌داری باشند.

4. وسایل نقلیه خودران: DQN می‌تواند برای تصمیم‌گیری در رانندگی خودران استفاده شود، جایی که agent یاد می‌گیرد در محیط‌های پیچیده حرکت کند.

نتیجه‌گیری

DQN یک الگوریتم قدرتمند است که نقاط قوت Q-learning و یادگیری عمیق را ترکیب کرده و به agentها امکان می‌دهد تا سیاست‌های مؤثر در محیط‌های پیچیده را یاد بگیرند. کاربردهای آن در حوزه‌های مختلف گسترده است، که آن را به ابزاری همه‌کاره در زمینه یادگیری تقویتی تبدیل کرده است.

منابع:

https://www.youtube.com/watch?v=x83WmvbRa2I

https://www.tensorflow.org/agents/tutorials/0_intro_rl

https://huggingface.co/learn/deep-rl-course/en/unit3/deep-q-algorithm

https://www.baeldung.com/cs/q-learning-vs-deep-q-learning-vs-deep-q-network

https://www.researchgate.net/figure/The-framework-of-deep-Q-network-DQN_fig3_349500153