بخش کتبی :

MDP

(1

$V_1((1,1)) = V_1((2,1)) = 0$

$V_1((2,2)) = \max\left[\underbrace{\left[0.8\times(0+0.9\times 5)+0\right]}_{a=R}, \underbrace{\left[0.1\times(0+0.9\times 5)+0\right]}_{a=D,\,a=U}\right]$

$= \max(3.6, 0.45) = 3.6$

$V_1((1,2)) = \max\left[\underbrace{\left[0.8\times(0-0.9\times 5)+0\right]}_{a=R}, \underbrace{\left[0.1\times(0-0.9\times 5)\right]}_{a=U,\,a=D}, \underbrace{0}_{a=L}\right]$

$= \max(-3.6, -0.45, 0, -0.45) = 0$

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

$V_2((1,1)) = \max(0,0,0,0) = 0$

$V_2((2,2)) = 3.924 \qquad V_2((1,2)) = 2.142$

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)
$\pi^*(s)$	U	L	-	R	R	-

3)

$$V((1,1)) = \frac{1}{N((1,1))} \sum_{i=1}^{N((1,1))} G_i = \frac{1}{3}(5+5+5) = \frac{5}{3}$$

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

4) با توجه به اینکه agent انتخاب کرده که همیشه به راست برود و فرمول TD-Learning که به شرح زیر است :

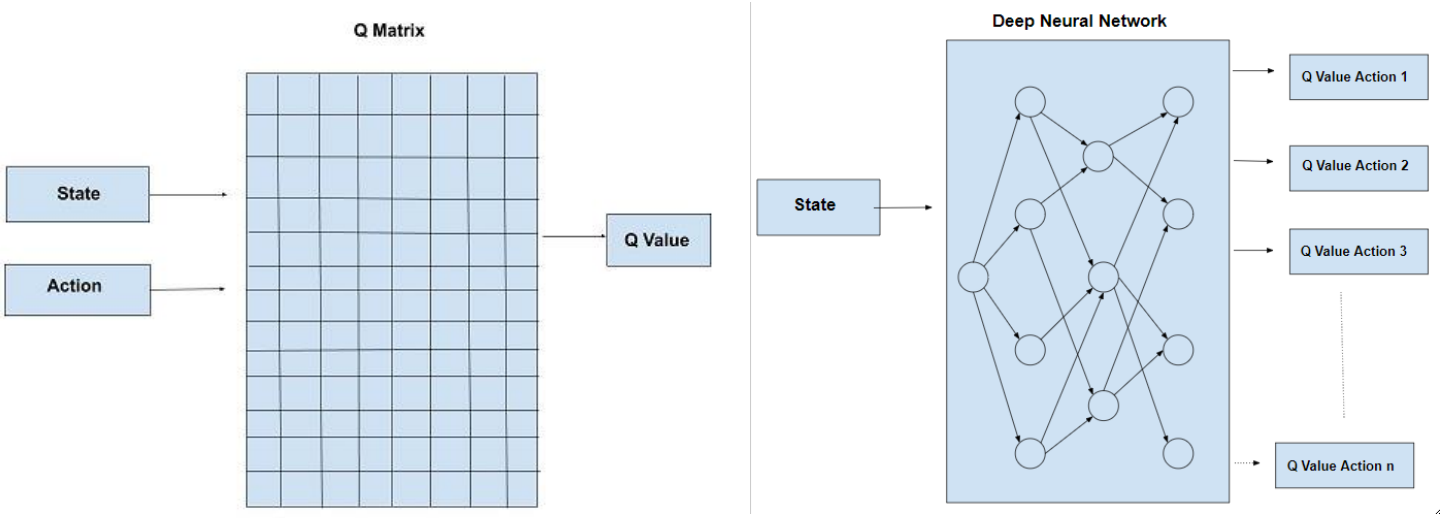
$$\alpha = 0.1 \qquad V^{\pi}(s) \leftarrow (1-\alpha)V^{\pi}(s) + \alpha[R(s,\pi(s),s') + V^{\pi}(s')]$$

مقادیر جدول بعد از هر iteration

S	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)
v_0^{π}	0	0	**-5**	0	0	**5**
v_1^{π}	0	-0.5	**-5**	0	0.5	**5**
v_2^{π}	-0.05	-0.95	**-5**	0.05	0.95	**5**

Deep Q-Networks

از ترکیب شبکه‌های عصبی عمیق با Q-Learning استفاده کرده تا الگوهای پیچیده‌تری را یاد بگیرد. در زیر مقایسه این مدل با Q-Table دیده می‌شود. معرفی شده توسط DeepMind در سال 2013



این شیوه به خصوص در محیط‌هایی با **تعداد حالات زیاد** کاربرد دارد

نکات مثبت **DQN** :

- مناسب برای محیط‌هایی با پیچیدگی زیاد (به دلیل کمک گرفتن از شبکه‌های عصبی عمیق)
- توانایی جنرالیزیشن برای استفاده در حالت‌هایی که از قبل دیده نشده
- کارایی بالاتر با نمونه‌های کمتر به دلیل استفاده از تکنیک Experience replay

محدودیت‌های **DQN** :

- حساس به مقادیر هایپرپارامترها و در نتیجه نیاز به آزمون و خطا هنگام تمرین مدل
- عدم توانایی continual learning به صورت طبیعی (آموزش روی نمونه‌های به صورت لایو و real-time)
- احتمال برآورد بیش‌ازحد برای ضرایب به دلیل نحوه محاسبه Q-Value

	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

بخش عملی :

هایپرپارامتر های استفاده شده

Learning Rate : تاثیر هر اکشن جدید و نتایج آن بر تغییر مقادیر Q-Value در هنگام آپدیت Q-Table

Discount Factor : کاهش/افزایش تاثیر اکشن‌های قدیمی‌تر نسبت به اکشن‌های جدیدتر در یک دنباله از اتفاقات هنگام آپدیت Q-Table

Epsilon : تنظیم نسبت Exploration در برابر Exploitation (گشتن به دنبال راه جدید یا تکیه بر بهترین اکشنی که قبلا پیدا کردیم)

Epsilon Decay : تنظیم میزان افت Epsilon بعد از هر Episode چون بهتر است که در ابتدای آموزش بیشتر به دنبال Exploration باشیم

Min Epsilon : کمترین میزان قابل قبول Epsilon که اگر در نتیجه Decay مقدار آن کمتر شود، به جای آن این مقدار قرار می‌گیرد تا هیچوقت کاملا 0 نشود (یکی از راه‌های جلوگیری از گیرکردن در Loop)

Observation Space :

بُعد اول : 8 مقدار – نشان دهنده مکان snack نسبت به سر مار

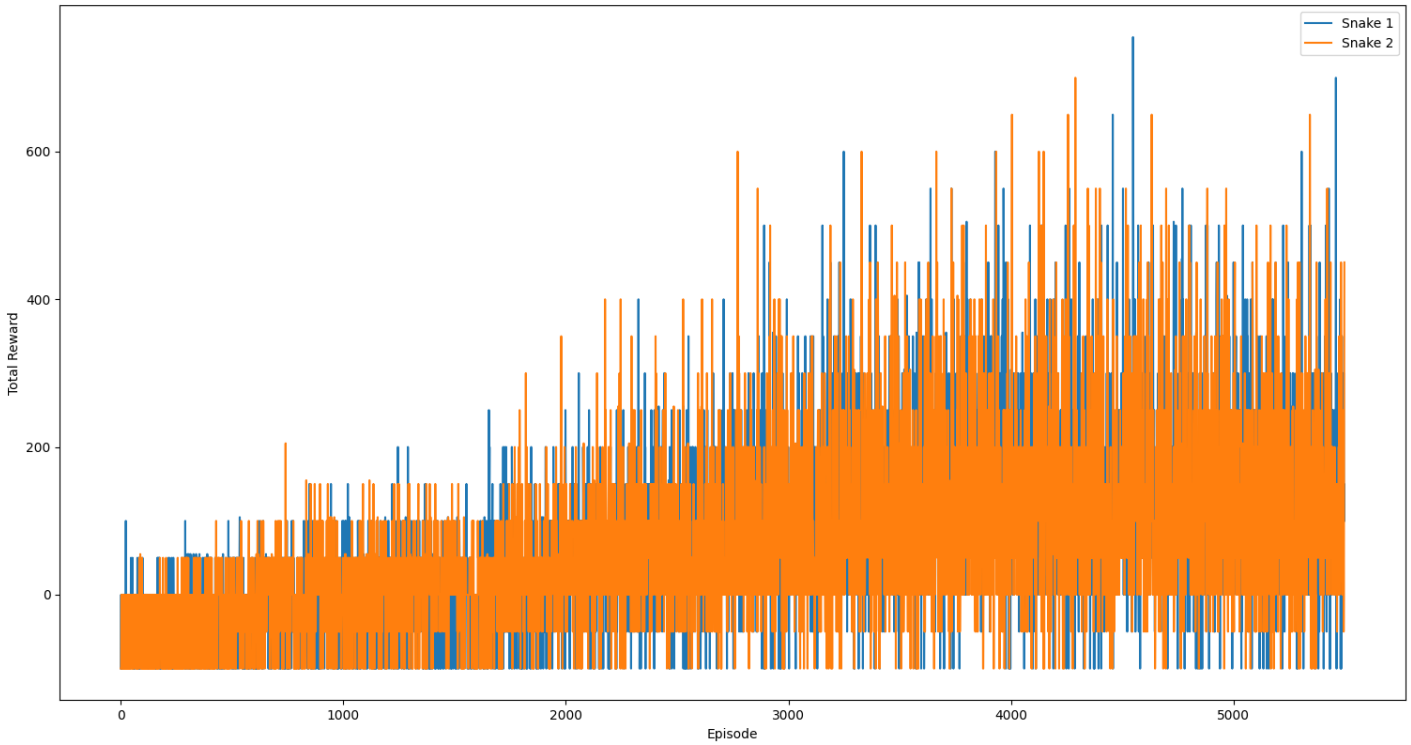
بعد دوم : 16 مقدار – نشان دهنده وجود خطر در 4 جهتی که سر میتواند به آن سو حرکت کند (در هر جهتی خط هست یا نیست پس در کل $2^4 = 16$ حالت

Rewards :

-100	خروج از برد / ضربه به خود / ضربه به بدن دیگری / ضربه به سر دیگری وقتی خودش کوتاه‌تر
(40, 50, 60)	رسیدن به *snack* (متغیر از مدل به مدل دیگر)
(0, -1, +1)	ضربه به سر دیگری با طول برابر (خاتمه بدون برنده)
(100, 10, 5)	ضربه به سر دیگری وقتی خودش بزرگتر

تحلیل و نمودار **مدل پنجم** از بین شش مدل ضمیمه شده :

میزان *Reward* دریافتی هر مار (مدل یکسان) بر حسب *Episode*



همانطور که مشاهده می‌شود در ابتدا به دلیل *Epsilon* بسیار زیاد مدل صرفا به صورت تصادفی درحال کشف محیط اطرافش است و مسیریابی واقعی انجام نمی‌دهد. اما از جایی که *Epsilon* به کمتر از 0.5 میل پیدا می‌کند تازه نمودار ما معنی‌دار می‌شود.

همچنین مقدار اختلاف کلی بین دقت دو مار دیده می‌شود با وجود اینکه هردو از یک مدل پیروی می‌کنند.

Observation Space :