.۱

در این مقاله یک معماری شبکه عصبی برای یادگیری model-free ارائه می‌شود. معماری Dueling می‌تواند بدون نیاز به یادگیری تأثیر هر عمل برای هر حالت ، بیاموزد که کدام حالت‌ها با ارزش هستند (یا نیستند). این امر به ویژه در حالت‌هایی که عمل‌های آن به هیچ وجه بر محیط تأثیر نمی گذارد بسیار مفید است.

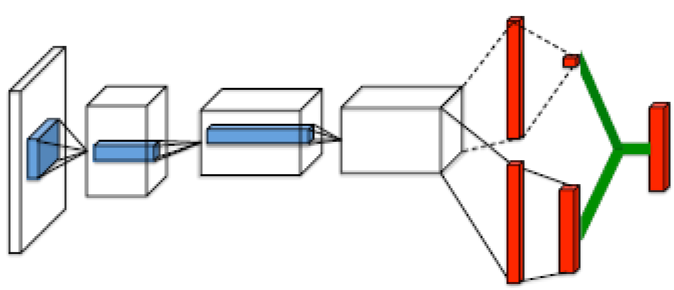
در بسیاری از حالت‌ها، برآورد ارزش هر عمل انتخابی ضروری نیست. در برخی از حالت‌ها ، دانستن اینکه چه عملی باید انجام شود از اهمیت فوق‌العاده‌ای برخوردار است ، اما در بسیاری از حالت‌های دیگر انتخاب عمل هیچ تأثیری در آنچه اتفاق می افتد ندارد.

برای ساخت این معماری سه تابع Q ، V (Value) و Advantage را برای یک سیاست (Policy) π مطابق زیر تعریف می‌کنیم:

$$\begin{aligned} Q^{\pi}(s,a) &= \mathbb{E}\left[R_t | \, s_t = s, a_t = a, \pi\right], \\ V^{\pi}(s) &= \mathbb{E}_{a\sim\pi(s)}\left[Q^{\pi}(s,a)\right]. \end{aligned}$$

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s).$$

تابع Value (V) مقدار خوب بودن یک حالت خاص s را اندازه‌گیری می‌کند. تابع Q ، ارزش انتخاب یک عمل خاص را هنگام قرار گرفتن در این حالت اندازه گیری می‌کند. تابع Advantage (A) مقدار Value حالت را از Q کم می کند تا اندازه نسبی اهمیت هر عمل را بدست آورد. می‌باشد.



لایه‌های پایین شبکه Dueling از نوع کانوولشنی هستند. بعد از لایه‌های کانوولشنی دو دنباله یا جریان از لایه‌های fully connected آورده می‌شوند. این دو دنباله تخمین یا برآوردهای جداگانه‌ای از توابع

Advantage و Value را به ما می‌دهند. در نهایت این دو دنباله را ترکیب می‌کنیم تا تابع Q را به دست آوریم. از روش‌های زیر برای ترکیب این دو تابع می‌توانیم استفاده کنیم:

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha), \qquad (7)$$

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \max_{a' \in |\mathcal{A}|} A(s,a';\theta,\alpha)\right). \qquad (8)$$

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a';\theta,\alpha)\right). \qquad (9)$$

مشکل روش اول این است که با داشتن Q نمی‌توانیم V و A را به صورت منحصر به فرد بازیابی کنیم بنابراین روش دوم پیشنهاد می‌شود. یک جایگزین برای روش دوم (8) روش سوم (9) است که البته باعث می‌شود V و A معنای اصلی خود را ار دست بدهند اما باعث پایداری (stability) فرایند optimization می‌شود زیرا در روش سوم Advantage فقط باید به سرعت میانگین تغییر کنند ، به جای اینکه مجبور شود هر تغییری را در Advantage عمل بهینه درروش دوم جبران کند.

از آنجا که خروجی شبکه Dueling یک تابع Q است ، می توان آن را با بسیاری از الگوریتم های موجود مانند DDQN و SARSA آموزش داد. علاوه بر این ، می‌توان از هرگونه بهبود در این الگوریتم ها ، از جمله حافظه‌های بازپخش (replay) بهتر ، سیاست های اکتشاف بهتر و غیره استفاده کرد.

۲.

Q-Learning:

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha(r_{t+1} + \gamma max_a Q(s_{t+1},a) - Q(s_t,a_t))$$

SARSA:

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha(r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t))$$

Q-Learning:

Q(State1, Up) = 2 + 0.9 * (-1 + 0.8 * 4 – 2) = 2.18

Q(State2, Right) = 4 + 0.9 * (-1 + 0.8 * 6 – 4) = 3.82

Q(State3, Right) = 6 + 0.9 * (-1 + 0.8 * 10 – 6) = 6.9

Q(State4, Down) = -40 + 0.9 * (-100 + 0.8 * 0 – (-40)) = -94

SARSA:

Q(State1, Up) = 2 + 0.9 * (-1 + 0.8 * 4 – 2) = 2.18

Q(State2, Right) = 4 + 0.9 * (-1 + 0.8 * 6 – 4) = 3.82

Q(State3, Right) = 6 + 0.9 * (-1 + 0.8 * (-40) – 6) = -29.1

Q(State4, Down) = -40 + 0.9 * (-100 + 0.8 * 0 – (-40)) = -94

.۳

$$V(S_t) \leftarrow V(S_t) + \alpha[G_t - V(S_t)]$$

$$G_t = R_{t+1} + \gamma\, R_{t+2} + \gamma^2\, R_{t+3} + \cdots$$

first -visit:

0	0	0	0	0
0	0	0	0	0
0	0		0	0
0	0		5	5
0	0	2.5	2.5	0

every-visit:

0	0	0	0	0
0	0	0	0	0
0	0		0	0
0	0		5	5
0	0	3.75	3.75	0

برای حالت‌هایی که در اپیزود نبودند G و متعاقب آن V صفر می‌ماند. با توجه به rewardهای حالت‌ها تا قبل از حالت 21 ، G برابر حاصل جمع تعدادی صفر و دوتا 5- و 10 بوده بنابراین G صفر می‌شود و و متعاقب آن V صفر می‌ماند. حالت 23 هم terminal هست بنابراین V آن از صفر تغییری نمی‌کند. برای حالت‌های 21، 22، 17، 18 با توجه به فرمول‌های V و G و روش‌های first-visit و every-visit مقادیر فوق در جدول به دست آمده‌اند.

first-visit:

$G_{21} = (0 + (-5) + 0 + 0 + 0 + 10) = 5$

$V(S_{21}) = 0 + 0.5 * (5 – 0) = 2.5$

$G_{22} = ((-5) + 0 + 0 + 0 + 10) = 5$

$V(S_{22}) = 0 + 0.5 * (5 – 0) = 2.5$

$G_{17} = (0 + 10) = 10$

$V(S_{17}) = 0 + 0.5 * (10 – 0) = 5$

$G_{18} = (10) = 10$

$V(S_{18}) = 0 + 0.5 * (10 – 0) = 5$

every-visit:

$G_{21} = ((0 + (-5) + 0 + 0 + 0 + 10) + (0 + 0 + 0 + 10)) / 2 = 7.5$

$V(S_{21}) = 0 + 0.5 * (7.5 – 0) = 3.75$

$G_{22} = (((-5) + 0 + 0 + 0 + 10) + (0 + 0 + 10)) / 2 = 7.5$

$V(S_{22}) = 0 + 0.5 * (7.5 – 0) = 3.75$

$G_{17} = (0 + 10) = 10$

$V(S_{17}) = 0 + 0.5 * (10 – 0) = 5$

$G_{18} = (10) = 10$

$V(S_{18}) = 0 + 0.5 * (10 – 0) = 5$

۴.

این سوال فقط برای ۱۰۰ اپیزود اجرا شده و با اینکه در این مدت جواب خوبی به دست نیامده اما گرفتن جواب خوب زمان زیادی می‌خواهد.