

ctionmark[۱]

۶۳۴ روش های بازیگر-منتقدalgorithm۶.



دانشگاه صنعتی شریف
دانشکده‌ی علوم ریاضی

پایان‌نامه‌ی کارشناسی ارشد
رشته علوم کامپیوتر

عنوان:

دسته بندی الگوریتم های یادگیری تقویتی

نگارش:

حسین یوسفی زاده

استاد راهنما:

دکتر دانشگر

دی ۱۳۹۹

سلام افلا

به نام خدا
دانشگاه صنعتی شریف
دانشکده‌ی علوم ریاضی

پایان‌نامه‌ی کارشناسی ارشد

عنوان: دسته بندی الگوریتم های یادگیری تقویتی
نگارش: حسین یوسفی زاده

کمیته‌ی ممتحنین

استاد راهنما: دکتر دانشگر
امضاء:

استاد مشاور: استاد مشاور
امضاء:

استاد مدعو: استاد ممتحن
امضاء:

تاریخ:

چکیده

تئوری یادگیری تقویتی (RL) به تدریج به یکی از فعال ترین حوزه های تحقیقاتی در یادگیری ماشین و هوش مصنوعی تبدیل شده، که ریشه در دیدگاه های روانشناختی و علوم اعصاب درباره رفتار حیوانات و انسان دارد. یادگیری تقویتی تلاش می کند به این سوال پاسخ دهد: چه کار کنیم که بیشترین پاداش یا کمترین هزینه نصیبمان شود؟ اینکه چگونه عوامل هوشمند می توانند کنترل خود را روی محیط بهینه کنند نیز در حوزه یادگیری تقویتی قرار می گیرد. عوامل هوشمند با کاری دشوار روبرو می شوند: آنها باید مدل های کارآمدی از محیط را با استفاده از ورودی های حسی بدست آورند و از این مدل ها برای تعمیم تجربه گذشته به موقعیت های جدید استفاده کنند. به نظر می رسد انسان و سایر حیوانات این مشکل را از طریق ترکیب هماهنگ یادگیری تقویتی و سیستم های حسی سلسله مراتبی حل می کنند. در این پایان نامه، به بررسی روش های مبتنی بر یادگیری تقویتی در حالت گسسته می پردازیم و الگوریتم های مهم آن را خواهیم دید. نهایتاً چند کاربرد دیگر را بررسی خواهیم کرد.

کلیدواژه ها: یادگیری تقویتی

فهرست مطالب

۸	۱ مقدمه
۹	۱-۱ تعریف مسئله
۹	۲-۱ اهمیت موضوع
۹	۳-۱ فضای حالت‌ها و عمل‌ها
۹	۴-۱ خط مشی
۹	۵-۱ سیگنال پاداش
۱۰	۶-۱ تابع ارزش
۱۰	۷-۱ محیط
۱۰	۸-۱ مسئله اکتشاف و بهره برداری
۱۰	۹-۱ اهداف تحقیق
۱۰	۱۰-۱ ساختار پایان‌نامه
۱۱	۲ مفاهیم اولیه یادگیری تقویتی
۱۲	۱-۲ دینامیک عامل-محیط
۱۲	۲-۲ فرایند تصمیم‌گیری مارکوف
۱۴	۳-۲ عناصر اصلی یادگیری تقویتی

۱۴ ۲-۳-۱ خطمشی
۱۴ ۲-۳-۲ سیگنال پاداش
۱۵ ۲-۳-۳ عایدی و تابع ارزش
۱۶ ۲-۳-۴ مدل محیط
۱۷ ۲-۴ برنامه‌ریزی پویا
۱۷ ۲-۴-۱ خطمشی و تابع ارزش بهینه
۱۸ ۲-۴-۲ معادله بلمن
۱۹ ۲-۴-۳ بهینگی و معادله بهینگی بلمن
۲۰ ۲-۴-۴ برنامه‌ریزی پویا
۲۱ ۲-۴-۵ بهبود خطمشی
۲۲ ۲-۴-۶ الگوریتم تکرار خطمشی
۲۳ ۲-۴-۷ الگوریتم تکرار ارزش
۲۴ ۲-۴-۸ درهم تنیدگی PE و PI و الگوریتم‌های GPI
۲۴ ۲-۵ روش‌های یادگیری تفاوت زمانی
۲۵ ۲-۵-۱ Q-learning
۲۶	۳ نتایج اخیر
۲۶ ۳-۱ روش‌های مبتنی بر مدل و بدون مدل
۲۷ ۳-۲ روش‌های بدون مدل
۲۷ ۳-۲-۱ روش‌های مبتنی بر ارزش
۳۱ ۳-۲-۲ روش‌های مبتنی بر خطمشی
۳۸ ۳-۲-۳ مقایسه روش بهینه‌سازی خطمشی و Q-learning
۳۹ ۳-۳ روش‌های مبتنی بر مدل

۳-۳-۱ روش مدل جهان ۳۹

آ مطالب تکمیلی ۴۰

فصل ۱

مقدمه

یک روش یادگیری ماشین تعریف می‌شود که مربوط به نحوه اقدام یک عامل هوشمند^۱ در محیط براساس هدفی مشخص است. یادگیری تقویتی عبارت است از یادگیری اینکه عامل هوشمند چه کاری باید انجام دهد (نحوه انتخاب اقدامات برحسب موقعیت) تا به حداکثر پاداش برسد. این روش تمامی مسائل یک عامل هدفمند را صریحاً در تعامل با یک محیط نامشخص بررسی می‌کند. به عامل هوشمند گفته نمی‌شود که چه کارهایی را انجام دهد، اما در عوض باید کشف کند که کدام اقدامات، بیشترین پاداش را به همراه دارد. در جالب ترین و چالش برانگیزترین موارد، اقدامات ممکن دارد نه تنها بر پاداش فوری بلکه در وضعیت بعدی محیط، و از طریق آن، بر کلیه پاداش های بعدی تأثیر بگذارد. این دو ویژگی (جستجوی آزمون و خطا و پاداش تأخیری) دو ویژگی مهم تمیز دهنده یادگیری تقویتی از روش های متداول یادگیری ماشین هستند. اقدامات عامل هوشمند، می‌تواند بر وضعیت آینده محیط تأثیر بگذارد یادگیری تقویتی یک رویکرد محاسباتی برای درک و خودکار کردن یادگیری و تصمیم‌گیری هدفمند است. یادگیری تقویتی، یادگیری از طریق تعامل است که چگونه می‌توان برای رسیدن به یک هدف رفتار کرد. عامل یادگیری تقویتی و محیط، در طی مراحل زمانی گسسته یا پیوسته با یکدیگر تعامل دارند.

¹ Agent

۱-۱ تعریف مسئله

مسئله‌ی یادگیری تقویتی در اصل یک مسئله بهینه سازی است. هدف اصلی مسئله، به حداکثر رساندن پاداشی است که از محیط دریافت می‌شود

تعریف دقیق‌تر این مسئله را در فصل دوم خواهیم دید.

۲-۱ اهمیت موضوع

یادگیری تقویتی در بسیاری از رشته‌ها مانند نظریه بازی، نظریه کنترل، تحقیق در عملیات، نظریه اطلاعات، بهینه سازی مبتنی بر شبیه سازی، سیستم‌های چند عاملی، هوش انبوه و آمار مورد مطالعه قرار می‌گیرد. در ادبیات تحقیق و کنترل عملیات، یادگیری تقویتی را برنامه ریزی تقریبی پویا^۲ یا برنامه ریزی عصبی پویا^۳ می‌نامند. مسائل مورد بررسی در یادگیری تقویتی در نظریه کنترل بهینه^۴ نیز مورد بررسی قرار گرفته است، که بیشتر مربوط به وجود و توصیف راه حل‌های بهینه و الگوریتم‌های محاسبه دقیق آنهاست، و کمتر مربوط به یادگیری یا تقریب، به ویژه در غیاب یک مدل ریاضی از محیط. در اقتصاد و نظریه بازی، ممکن است از یادگیری تقویتی برای توضیح چگونگی ایجاد تعادل، استفاده شود.

۳-۱ فضای حالت‌ها و عمل‌ها

۴-۱ خط مشی

۵-۱ سیگنال پاداش

terms in formalized is agent the of goal or purpose the learning, reinforcement In agent. the to environment the from passing reward, the called signal, special a of agent's the Informally, R_t . R_t number, simple a is reward the step, time each At

²Approximate Dynamic Programming

³Neuro-dynamic Programming

⁴Optimal Control Theory

maximiz- means This receives. it reward of amount total the maximize to is goal
clearly can We run. long the in reward cumulative but reward, immediate not ing
hypothesis reward the as idea informal this state

همه ی آنچه به عنوان هدف مدنظر داریم می تواند به صورت بیشینه سازی مقدار میانگین یک سیگنال
عددی بیان شود.

۱-۶ تابع ارزش

۱-۷ محیط

۱-۸ مسئله اکتشاف و بهره برداری

یکی از چالش هایی که در یادگیری تقویتی برخلاف سایر روش های یادگیری وجود دارد ، رقابت بین
اکتشاف و بهره برداری است. برای به دست آوردن پاداش زیاد ، یک عامل یادگیری تقویتی باید کارهایی
را ترجیح دهد که در گذشته انجام داده و در تولید پاداش موثرتر بوده است. اما برای کشف چنین اعمالی،
باید اقداماتی را امتحان کند که قبلاً انتخاب نکرده است. این عامل برای به دست آوردن پاداش مجبور
است از آنچه قبلاً تجربه کرده است بهره برداری کند، اما همچنین برای انتخاب اقدامات بهتر در آینده
باید به کاوش بپردازد. اکتشاف و بهره برداری هیچکدام به تنهایی در رسیدن به هدف، کارا نیست.

۱-۹ اهداف تحقیق

۱-۱۰ ساختار پایان نامه

این پایان نامه شامل — فصل است. در فصل ...

فصل ۲

مفاهیم اولیه یادگیری تقویتی

کنترل بهینه^۱ در اواخر دهه ۱۹۵۰ برای توصیف مسئله طراحی یک کنترل‌گر برای به حداقل رساندن اندازه‌گیری رفتار سیستم دینامیکی^۲ در طول زمان مورد استفاده قرار گرفت. یکی از رویکردهای این مسئله در اواسط دهه ۱۹۵۰ توسط ریچارد بلمن^۳ و دیگران از طریق گسترش نظریه قرن نوزدهم همیلتون^۴ و جاکوبی^۵ توسعه یافت. این رویکرد از مفاهیم حالت یک سیستم دینامیکی و یک تابع ارزش^۶ برای تعریف یک معادله تابعی استفاده می‌کند؛ که اکنون معادله بلمن^۷ نامیده می‌شود. مجموعه روش‌های حل مسائل کنترل بهینه به کمک معادله بلمن به عنوان برنامه‌ریزی پویا شناخته می‌شود. همچنین بلمن نسخه گسسته از مسئله کنترل بهینه را که تحت عنوان فرایندهای تصمیم‌گیری مارکوف^۸ شناخته می‌شود، معرفی کرد. رونالد هوارد (۱۹۶۰) روش Policy Iteration را برای MDP ها طراحی کرد. همه این‌ها عناصر اساسی در تئوری و الگوریتم‌های یادگیری تقویتی^۹ مدرن هستند. در این فصل، با فرایندهای تصمیم‌گیری مارکوف آشنا خواهیم شد و چهار عنصر اصلی یادگیری تقویتی، یعنی خط‌مشی^{۱۰}، سیگنال پاداش^{۱۱}، تابع ارزش و محیط^{۱۲} را دقیقاً تعریف خواهیم کرد. همچنین برخی روش‌های کلاسیک در

¹Optimal control

²Dynamical system

³Richard Bellman

⁴Hamilton

⁵Jacobi

⁶Value function

⁷Bellman equation

⁸Markov decision process(MDP)

⁹Reinforcement learning (RL)

¹⁰Policy

¹¹Reward signal

¹²Environment

یادگیری تقویتی را معرفی می‌کنیم.

۲-۱ دینامیک عامل-محیط

حوزه یادگیری تقویتی دو بازیگر اصلی دارد؛ عامل و محیط. موجود تصمیم گیرنده و آموزنده را عامل یادگیری، یا به اختصار، عامل می‌نامند. قسمتی که عامل با آن تعامل دارد (هر چیزی خارج از عامل)، محیط نامیده می‌شود. در ادبیات کنترل بهینه، معمولاً به جای واژه‌های عامل و محیط، از کنترل‌کننده^{۱۳} و سیستم کنترل شده^{۱۴} استفاده می‌شود. عامل و محیط به طور مداوم با یکدیگر ارتباط برقرار می‌کنند. عامل انتخاب می‌کند که چه اقدامی انجام دهد، محیط به این اقدامات پاسخ می‌دهد و موقعیت جدیدی را به عامل ارائه می‌دهد. محیط، همچنین مقادیر عددی ویژه ای به نام پاداش به عامل برمی‌گرداند، که عامل به دنبال به حداکثر رساندن آن است.

به طور خاص، عامل و محیط در یک توالی زمانی گسسته تعامل می‌کنند، $t = 0, 1, 2, 3, \dots$ ، در هر مرحله t ، عامل، وضعیت محیط $S_t \in \mathcal{S}$ را دریافت می‌کند، و بر اساس آن یک عمل $A_t \in \mathcal{A}$ را انتخاب می‌کند. در گام بعدی، عامل به عنوان نتیجه عمل خود، یک پاداش عددی $R_{t+1} \in \mathcal{R}$ دریافت می‌کند و خود را در حالت جدید S_{t+1} می‌یابد. دینامیک عامل-محیط را می‌توان به شکل یک دنباله از حالت‌ها، عمل‌ها و پاداش‌ها به صورت زیر نمایش داد:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \dots$$

۲-۲ فرایند تصمیم‌گیری مارکوف

یادگیری تقویتی از چارچوب رسمی فرایندهای تصمیم‌گیری مارکوف^{۱۵} (MDP) برای تعریف تعامل بین یک عامل یادگیری و محیط توسط حالت‌ها، اقدامات و پاداش استفاده می‌کند. مدل MDP یک مدل کلاسیک از تصمیم‌گیری متوالی است، جایی که اقدامات نه تنها بر پاداش‌های فوری، بلکه بر موقعیت‌ها

¹³Controller

¹⁴Controlled System

¹⁵Marcov Decision Process

و حالت‌های بعدی و به تبع آن بر پاداش‌های آینده تأثیر می‌گذارد. MDP یک فرم ایده‌آل ریاضی از مسئله یادگیری تقویتی است که برای آن تئوری‌های دقیقی بیان شده‌است. MDP متناهی، یک MDP با مجموعه حالت‌های متناهی است. بیشتر نظریه‌های فعلی یادگیری تقویتی، محدود به MDP های متناهی است، اما روش‌ها و ایده‌ها به طور کلی بیان می‌شوند.

تعریف ۱-۲ (فرایند تصمیم‌گیری مارکوف) فرایند تصمیم‌گیری مارکوف، یک ۴ تایی

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$$

است که

- \mathcal{S} بیانگر مجموعه تمام حالت‌هاست،
- \mathcal{A} بیانگر مجموعه تمام عمل‌هاست،
- $\mathcal{R} \subseteq \mathbb{R}$ بیانگر مجموعه پاداش‌هاست،

- \mathcal{P} هسته احتمال انتقال^{۱۶}، $\mathcal{P} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S} \times \mathcal{R})$ ، تابعی است که دینامیک MDP را مشخص می‌کند.

هسته احتمال انتقال یا تابع انتقال \mathcal{P} ، هر دوتایی حالت-عمل (s, a) که $s \in \mathcal{S}$ و $a \in \mathcal{A}$ ، را به یک توزیع احتمال روی دوتایی‌هایی به شکل (s', r) نسبت می‌دهد. s' بیانگر حالت بعدی و r بیانگر پاداش این انتقال است. به ازای هر دو حالت $s, s' \in \mathcal{S}$ و هر عمل $a \in \mathcal{A}$ و هر پاداش $r \in \mathcal{R}$ احتمال رسیدن به حالت s' و دریافت پاداش r با انتخاب عمل a در حالت s ، یک عدد حقیقی متعلق به بازه $[0, 1]$ است که آن را به شکل $p(s', r | s, a)$ نمایش می‌دهیم:

$$p(s', r | s, a) \triangleq \Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}.$$

نامگذاری فرایند تصمیم‌گیری مارکوف اشاره به این موضوع دارد که این سیستم‌ها دارای ویژگی مارکوف^{۱۷} هستند، بدین معنا که تابع انتقال آن تنها به حالت فعلی سیستم و آخرین عمل انجام شده وابسته است، و نسبت به حالت‌ها و اعمال قبلی مستقل است.

¹⁶Transition Probability Kernel

¹⁷Marcov Property

۳-۲ عناصر اصلی یادگیری تقویتی

چهار عنصر کلیدی و اصلی مسائل یادگیری تقویتی عبارت است از: خط‌مشی^{۱۸}، سیگنال پاداش^{۱۹}، تابع ارزش^{۲۰} و محیط^{۲۱}

۱-۳-۲ خط‌مشی

خط‌مشی^{۲۲} نحوه رفتار عامل یادگیری را در یک زمان خاص، مشخص می‌کند و هسته اصلی رفتار یک عامل یادگیری تقویتی است. به بیان دیگر، خط‌مشی، نگاشتی از حالت‌های مدل شده از محیط به عملی است که باید در آن حالت انجام شود. خط‌مشی به تنهایی برای تعیین رفتار عامل کافی است. ممکن است خط‌مشی عامل یادگیری، یک تابع ساده یا جدول جستجو باشد، یا ممکن است شامل محاسبات پیچیده‌ای مانند فرآیند جستجو باشد؛ همچنین خط‌مشی‌ها ممکن است احتمالاتی باشند.

تعریف ۲-۲ (خط‌مشی احتمالاتی ثابت) یک خط‌مشی احتمالاتی ثابت^{۲۳} (یا به طور خلاصه خط‌مشی ثابت) $\pi : S \rightarrow \Pi(A)$ تابعی است که هر حالت را به یک توزیع احتمال روی فضای عمل A می‌نگارد. برای سادگی، احتمال انتخاب عمل a در حالت s تحت خط‌مشی π را به شکل $\pi(a|s)$ نشان می‌دهیم.

تعریف ۳-۲ می‌گوییم خط‌مشی π در یک MDP دنبال می‌شود هرگاه

$$A_t \sim \pi(\cdot|S_t), \quad t \in \mathbb{N}.$$

۲-۳-۲ سیگنال پاداش

در مسئله یادگیری تقویتی، تصمیمات عامل یادگیری، توسط سیگنال پاداش جهت‌دهی می‌شود. در هر گام، محیط، یک عدد حقیقی به عنوان پاداش برای عامل یادگیری تقویتی ارسال می‌کند. تنها هدف عامل، به حداکثر رساندن کل پاداش دریافتی از محیط در طولانی مدت است. سیگنال پاداش اتفاقات

¹⁸Policy

¹⁹Reward Signal

²⁰Value Function

²¹Environment

²²Policy

²³Stationary probabilistic policy

خوب و بد را برای عامل مشخص می‌کند و مبنای اصلی تغییر خط‌مشی است سیگنال پاداش می‌تواند تابعی تصادفی از وضعیت محیط و عمل انجام شده باشد.

۲-۳-۳ عایدی و تابع ارزش

همانطور که سیگنال پاداش نشان می‌دهد که انجام چه عملی در هر گام خوب است، تابع ارزش مشخص می‌کند که کدام عمل در طولانی مدت بهتر است. در واقع تابع ارزش نشانگر مطلوبیت طولانی مدت حالت‌ها با در نظر گرفتن حالت‌هایی است که در پی خواهند داشت. ارزش حالت s ، تحت خط‌مشی π ، یا $v_{\pi}(s)$ ، مجموع میزان پاداشی است که عامل، با شروع از s و دنبال کردن خط‌مشی π ، می‌تواند انتظار داشته باشد در آینده کسب کند. پاداش‌ها به یک معنا اولیه هستند، در حالی که ارزش‌ها، به عنوان پیش‌بینی پاداش‌ها، ثانویه هستند. بدون پاداش هیچ ارزشی وجود ندارد و تنها هدف تخمین ارزش‌ها، دستیابی به پاداش بیشتر است. با این وجود، این تابع ارزش است که هنگام تصمیم‌گیری و ارزیابی به آن توجه می‌کنیم. تعیین ارزش، بسیار دشوارتر از تعیین پاداش است. ارزش‌ها باید از توالی مشاهداتی که یک عامل در طول عمر خود انجام می‌دهد، برآورد شوند. مهمترین مولفه بیشتر الگوریتم‌های یادگیری تقویتی که در این فصل و فصل آینده معرفی خواهیم کرد، روشی برای تخمین کارآمد تابع ارزش است.

تعریف ۲-۴ عایدی آینده کاهشی^{۲۴} یا به اختصار، عایدی^{۲۵}، در زمان t به شکل

$$G_t \triangleq \sum_{t'=t}^T \gamma^{t'-t} R_{t'}$$

تعریف می‌شود که T زمانی است که اپیزود به اتمام می‌رسد. اگر مسئله مستمر باشد آنگاه $T = \infty$

از تعریف بالا نتیجه می‌شود

$$\begin{aligned} G_t &= R_t + \sum_{t'=t+1}^T \gamma^{t'-t} R_{t'} \\ &= R_t + \gamma \sum_{t'=t+1}^T \gamma^{t'-(t+1)} R_{t'} \\ &= R_t + \gamma G_{t+1}. \end{aligned} \quad (1-2)$$

²⁴Future Discounted Return

²⁵Return

تعریف ۲-۵ (تابع ارزش حالت) ارزش حالت s تحت خط‌مشی π یا $v_\pi(s)$ به شکل امیدریاضی عایدی، با شروع از s و دنبال کردن خط‌مشی π تعریف می‌شود.

$$v_\pi(s) \triangleq \mathbb{E}_\pi [G_t | S_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right]$$

تابع v_π را تابع ارزش حالت ^{۲۶} مربوط به خط‌مشی π می‌نامیم. اگر s یک حالت نهایی باشد آنگاه $v_\pi(s) = 0$.

تعریف ۲-۶ (تابع ارزش عمل) ارزش عمل a در حالت s تحت خط‌مشی π یا $q_\pi(s, a)$ به شکل امیدریاضی عایدی، با شروع از s و انتخاب عمل a و سپس دنبال کردن خط‌مشی π تعریف می‌شود.

$$q_\pi(s, a) = \mathbb{E}_\pi [G_t | S_t = s, A_t = a] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right]$$

تابع q_π را تابع ارزش عمل ^{۲۷} مربوط به خط‌مشی π می‌نامیم.

۲-۳-۴ مدل محیط

مدل محیط رفتار محیط را تقلید می‌کند، یا به طور کلی‌تر، اجازه می‌دهد تا در مورد نحوه رفتار محیط، پیشبینی کارآمدی داشته‌باشیم. از مدل‌ها برای برنامه‌ریزی و انتخاب در روند تصمیم‌گیری با در نظر گرفتن شرایط احتمالی آینده بدون تجربه واقعی آن‌ها استفاده می‌شود. روش‌هایی که برای حل مشکلات یادگیری تقویتی از مدل‌ها و برنامه‌ریزی‌ها استفاده می‌کنند، روش‌های مبتنی بر مدل ^{۲۸} نامیده می‌شوند. در مقابل این روش‌ها، روش‌های بدون مدل ^{۲۹} هستند که از هیچ گونه شبیه‌سازی یا ابزاری برای پیشبینی رفتار محیط استفاده نمی‌کنند و معمولاً روند یادگیری آن‌ها بر اساس تجربه واقعی در محیط و آزمون و خطا است.

²⁶State Value Function

²⁷Action Value Function

²⁸Model Based

²⁹Model Free

۴-۲ برنامه‌ریزی پویا

در بخش قبل، با مفاهیم کلیدی یادگیری تقویتی آشنا شدیم. در این بخش به یکی از مهم‌ترین روش‌های کلاسیک در حل مسئله یادگیری تقویتی، یعنی برنامه‌ریزی پویا^{۳۰} خواهیم پرداخت. برنامه‌ریزی پویا مجموعه‌ای از الگوریتم‌ها است در صورت وجود مدل کاملی از محیط در قالب یک MDP می‌توانند برای محاسبه بهترین خط‌مشی (خط‌مشی‌ای که بیشترین عایدی را به دست می‌دهد) استفاده شوند. الگوریتم‌های کلاسیک برنامه‌ریزی پویا به دلیل فرض مدل کاملی از محیط و همچنین هزینه محاسباتی زیادشان، به لحاظ عملی چندان قابل استفاده نیستند اما به لحاظ نظری مهم هستند. قبل از آن که به روش‌های برنامه‌ریزی پویا بپردازیم، لازم است با مفهوم خط‌مشی بهینه و تابع ارزش بهینه آشنا شویم.

۱-۴-۲ خط‌مشی و تابع ارزش بهینه

حل کردن مسئله یادگیری تقویتی، به معنی پیدا کردن خط‌مشی‌ای است که بیشترین پاداش را در طول زمان موجب می‌شود. به چنین خط‌مشی‌ای، خط‌مشی بهینه^{۳۱} گفته می‌شود. برای MDP های متناهی، می‌توانیم خط‌مشی بهینه را به صورت زیر تعریف کنیم:

تعریف ۷-۲ می‌گوییم خط‌مشی π بهتر یا مساوی خط‌مشی π' است یا $\pi \geq \pi'$ هرگاه برای هر $s \in S$

$$v_{\pi}(s) \geq v_{\pi'}(s).$$

می‌توان نشان داد که برای هر MDP متناهی، حداقل یک خط‌مشی وجود دارد که بهتر یا مساوی هر خط‌مشی دیگری باشد [۱]. به چنین خط‌مشی‌ای خط‌مشی بهینه گفته می‌شود. ممکن است بیش از یک خط‌مشی بهینه وجود داشته باشد. تمام خط‌مشی‌های بهینه را با نماد π_* نمایش می‌دهیم. تابع ارزش متناظر با همه خط‌مشی‌های بهینه یکسان است و برابر با تابع ارزش بهینه^{۳۲} است که با نماد v_* نمایش داده شده و به شکل زیر تعریف می‌شود:

$$v_*(s) \triangleq \max_{\pi} v_{\pi}(s).$$

³⁰ (DP)Dynamic Programming

³¹Optimal Policy

³²Optimal Value Function

همچنین تمام خط‌مشی‌های بهینه تابع ارزش عمل مشترکی دارند که آن را با نماد q_* نمایش می‌دهیم و به شکل زیر تعریف می‌کنیم:

$$q_*(s, a) \triangleq \max_{\pi} q_{\pi}(s, a).$$

می‌توانیم q_* را برحسب v_* به شکل زیر بنویسیم

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a]. \quad (2-2)$$

۲-۴-۲ معادله بلمن

ویژگی اساسی توابع ارزش که در طول یادگیری تقویتی و برنامه‌ریزی پویا استفاده می‌شوند، این است که آن‌ها در یک رابطه بازگشتی موسوم به معادله بلمن^{۳۳} صدق می‌کنند. معادله بلمن رابطه‌ای بین ارزش یک حالت و ارزش‌های حالت‌های بعدی آن را بیان می‌کند.

فرض کنید π یک خط‌مشی دلخواه باشد و $s \in \mathcal{S}$. با استفاده مستقیم از تعریف ۲-۵ داریم

$$\begin{aligned} v_{\pi}(s) &\triangleq \mathbb{E}_{\pi}[G_t | S_t = s] \\ &= \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma \mathbb{E}_{\pi}[G_{t+1} | S_{t+1} = s']] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) [r + \gamma v_{\pi}(s')]. \end{aligned} \quad (3-2)$$

به رابطه ۲-۳ معادله بلمن گفته می‌شود. این معادله، رابطه‌ای بین ارزش یک حالت و ارزش حالت‌های بعدی آن را مشخص می‌کند. مشابه این رابطه را می‌توان برای تابع ارزش عمل نیز تحقیق کرد. فرض کنید $s \in \mathcal{S}$ و $a \in \mathcal{A}$. داریم

³³Bellman Equation

$$\begin{aligned}
q_\pi(s, a) &\triangleq \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\
&= \sum_{s', r} p(s', r | s, a) \sum_{a'} \pi(a' | s') [r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s', A_{t+1} = a']] \\
&= \sum_{s', r} p(s', r | s, a) \sum_{a'} \pi(a' | s') [r + \gamma q_\pi(s', a')] \quad \forall s \in (S) \tag{۴-۲}
\end{aligned}$$

رابطه ۲-۴ به معادله بلمن مربوط به تابع ارزش عمل معروف است.

۳-۴-۲ بهینگی و معادله بهینگی بلمن

از آنجا که v_* تابع ارزش یک خط‌مشی است، بنابراین باید در شرایط معادله ۲-۳ صدق کند. در این حالت خاص، معادله بلمن را می‌توان به فرم ویژه‌ای نوشت که با نام معادله بهینگی بلمن^{۳۴} شناخته می‌شود.

$$\begin{aligned}
v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s, a) \\
&= \max_a \mathbb{E}_{\pi_*}[G_t | S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \\
&= \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma v_*(s')]. \tag{۵-۲}
\end{aligned}$$

معادله ۲-۵ بیانگر این واقعیت است که ارزش یک حالت تحت یک خط‌مشی بهینه، برابر با امیدریاضی عایدی، برای بهترین عمل از آن حالت است. به طور مشابه برای تابع ارزش عمل بهینه، معادله بهینگی بلمن به شکل زیر خواهد بود:

$$\begin{aligned}
q_*(s, a) &= \mathbb{E} \left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a \right] \\
&= \sum_{s', r} p(s', r | s, a) \left[r + \gamma \max_{a'} q_*(s', a') \right]. \tag{۶-۲}
\end{aligned}$$

³⁴Bellman Optimality Equation

۲-۴-۲ برنامه‌ریزی پویا

ایده اصلی DP و به طور کلی یادگیری تقویتی، استفاده از تابع ارزش حالت یا عمل برای سازماندهی یک الگوریتم جستجو برای پیدا کردن خط‌مشی بهینه است. v_* یا q_* است که در معادلات بهینگی بلمن صدق می‌کند:

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')] \quad (۷-۲)$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')] \quad (۸-۲)$$

اگر دینامیک محیط کاملاً معلوم باشد معادله ۲-۷ و یک دستگاه معادلات خطی با اندازه $|S|$ است که راه‌حل آن سراسر است. اما اگر تعداد حالت‌ها زیاد پاید، ممکن است این روش برای اهداف ما، عملی نباشد. روش‌های تکراری مناسب‌ترین روش‌ها هستند. یک دنباله v_0, v_1, v_2, \dots از توابع ارزش تقریبی را در نظر بگیرید که هرکدام نگاشتی از S به \mathbb{R} هستند. تقریب اولیه v_0 به طور دلخواه انتخاب می‌شود (به جز در حالت‌های پایانی که باید صفر باشد). هر تقریب به عنوان یک قانون بروزرسانی به وسیله معادله بلمن از روی تقریب قبلی برای v_π بدست می‌آید.

$$\begin{aligned} v_{k+1}(s) &\triangleq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')] \end{aligned} \quad (۹-۲)$$

برای هر $s \in S$. بر اساس معادله بلمن می‌توان درستی این تساوی را برای v_π نوشت و واضح است که $v_k = v_\pi$ نقطه ثابتی برای این قانون بروزرسانی است. در واقع، می‌توان نشان داد که در حالت کلی دنباله $\{v_k\}$ وقتی که $k \rightarrow \infty$ و تحت همان شرایط که وجود v_π را تضمین می‌کند، به مقدار v_π همگرا می‌شود. این الگوریتم را ارزیابی خط‌مشی گام به گام ^{۳۵} یا به طور خلاصه ارزیابی خط‌مشی ^{۳۶} می‌نامند.

^{۳۵}Iterative policy evaluation^{۳۶}Policy Evaluation

الگوریتم ۱ الگوریتم ارزیابی خط مشی

ورودی: خط مشی π

۱: تا رسیدن به همگرایی تکرار کن:

۲: برای هر $s \in \mathcal{S}$:

$$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) [r + \gamma V(s')] \quad ۳:$$

خروجی: $V \approx v_\pi$

۲-۴-۵ بهبود خط مشی

هدف ما برای محاسبه تابع ارزش یک خط مشی کمک به یافتن خط مشی های بهتر است. فرض کنید ما تابع ارزش v_π را برای یک خط مشی معین^{۳۷} دلخواه π محاسبه کرده ایم. برای یک حالت s می خواهیم بدانیم که آیا باید خط مشی را برای انتخاب قطعی یک عمل $a \neq \pi(s)$ تغییر دهیم یا خیر. ما می دانیم که پیروی از خط مشی فعلی از حالت s چقدر خوب است، اما آیا تغییر به یک خط مشی جدید بهتر است یا بدتر؟ یکی از راه های پاسخ به این سوال در نظر گرفتن انتخاب عمل a در حالت s و پس از آن پیروی از خط مشی π است. ارزش این شیوه عملکرد برابر است با:

$$q_\pi(s, a) = \sum_{s',r} p(s', r|s, a) [r + \gamma v_\pi(s')]$$

اگر این عبارت از $v_\pi(s)$ بزرگتر باشد، بدین معناست که انتخاب قطعی عمل a در حالت s و سپس دنبال کردن خط مشی π بهتر از این است که همواره خط مشی π را دنبال کنیم. این گزاره را می توان به فرم کلی تر در قالب یک قضیه بیان کرد.

قضیه ۱-۲ (قضیه بهبود خط مشی) فرض کنید π و π' دو خط مشی معین باشند که برای هر $s \in \mathcal{S}$

$$q_\pi(s, \pi'(s)) \geq \pi(s) \quad (۱۰-۲)$$

در این صورت

$$v_{\pi'}(s) \geq v_\pi(s) \quad (۱۱-۲)$$

^{۳۷}Deterministic

همچنین اگر نامساوی ۲-۱۰ برای یکی از حالت ها، به صورت اکید برقرار باشد آنگاه ۲-۱۱ نیز به صورت اکید برقرار است.

درستی قضیه بالا را می توان با استفاده از تعاریف به روشنی بررسی کرد:

$$\begin{aligned}
 v_{\pi}(s) &\leq q_{\pi}(s, \pi'(s)) = \mathbb{E} [R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s, A_t = \pi'(a)] \\
 &= \mathbb{E}_{\pi'} [R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s] \\
 &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma q_{\pi}(S_{t+1}, \pi'(S_{t+1})) | S_t = s] \\
 &= \mathbb{E}_{\pi'} [R_{t+1} + \gamma \mathbb{E}_{\pi'} [R_{t+2} + \gamma v_{\pi}(S_{t+2})] | S_t = s] \\
 &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^2 v_{\pi}(S_{t+3}) | S_t = s] \\
 &\vdots \\
 &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \dots | S_t = s] \\
 &\leq v_{\pi'}(s)
 \end{aligned}$$

الگوریتم ۲ الگوریتم بهبود خط مشی

ورودی: خط مشی π و تابع $V \approx v_{\pi}$

۱: برای هر $s \in \mathcal{S}$:

۲: $\pi(s) \leftarrow \arg \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V(s')]$

خروجی: $V \approx v_{\pi}$

۲-۴-۶ الگوریتم تکرار خط مشی

قضیه ۲-۱ یک شرط کافی برای اینکه خط مشی π' بهتر یا مساوی خط مشی π باشد را بیان می کند. این قضیه می تواند ما را به روشی برای بهبود یک خط مشی، سوق دهد. منظور ما از بهبود خط مشی، روشی برای دستیابی به یک خط مشی بهتر، از روی یک خط مشی ضعیف تر است.

فرض کنید π' خط مشی معینی باشد که نسبت به تابع ارزش عمل q_{π}

$$\begin{aligned}
\pi'(s) &\triangleq \arg \max_a q_\pi(s, a) \\
&= \arg \max_a \mathbb{E} [R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a] \\
&= \arg \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma v_\pi(s')].
\end{aligned}$$

با استفاده از قضیه ۲-۱ می توان نشان داد

$$v_{\pi'}(s) \geq v_\pi(s).$$

بنابراین با استفاده از تابع ارزش خط مشی π می توان به یک خط مشی بهتر رسید [۱]. می توانیم با استفاده از ارزیابی خط مشی، روی π' به تخمینی از $v_{\pi'}$ برسیم؛ سپس می توانیم $v_{\pi'}$ را محاسبه کرده و مجدداً آن را بهبود بخشیم تا خط مشی بهتری داشته باشیم. با ادامه دادن این روند، می توانیم دنباله ای از خط مشی ها و تابع ارزش ها به دست آوریم که به صورت یکنوا در حال بهبود هستند:

$$\pi_0 \longrightarrow v_{\pi_0} \longrightarrow \pi_1 \longrightarrow v_{\pi_1} \longrightarrow \pi_2 \longrightarrow \dots \longrightarrow \pi_* \longrightarrow v_*$$

می توان نشان داد که این دنباله از خط مشی ها و توابع ارزش، به خط مشی و تابع ارزش بهینه همگرا می شود. به این روش، روش تکرار خط مشی^{۳۸} گفته می شود.

۲-۴-۷ الگوریتم تکرار ارزش

یک اشکال در روش تکرار خط مشی این است که هر یک از تکرارهای آن، شامل ارزیابی خط مشی است که به خودی خود یک محاسبه زمان بر است که نیاز به رفت و برگشت های متعدد روی مجموعه حالت دارد. اگر ارزیابی خط مشی بصورت گام به گام تکراری انجام شود، همگرایی فقط در حد اتفاق می افتد. بنابراین یا باید منتظر همگرایی دقیق بمانیم و یا می توانیم قبل تر از آن متوقف شویم. در واقع، عملیات ارزیابی خط مشی را می توان پس از یک بار رفت و برگشت روی فضای حالت (یک بروزرسانی در هر حالت)، بدون از دست دادن تضمین همگرایی به خط مشی بهینه، متوقف کرد. این الگوریتم، تکرار ارزش^{۳۹} نامیده می شود. این الگوریتم را می توان به عنوان یک عملیات بروزرسانی ساده نوشت که ترکیبی از بهبود خط مشی و مراحل ارزیابی خط مشی کوتاه شده است.

³⁸Policy iteration

³⁹Value Iteration

۲-۴-۸ درهم تنیدگی PE و PI و الگوریتم‌های GPI

یک اشکال عمده در روشهای DP که تا کنون بحث شده است، این است که آنها شامل عملیات‌هایی روی تمام مجموعه حالت‌ها هستند. اگر فضای حالت بسیار بزرگ باشد، حتی یک بار رفت و برگشت می‌تواند بسیار پرهزینه باشد. الگوریتم‌های ناهمزمان DP الگوریتم‌های تکرار شونده درجا هستند که مقید به رفت و برگشت سیستماتیک روی تمام مجموعه حالت‌ها نیستند. این الگوریتم‌ها ارزش حالت‌ها را بدون هیچ ترتیب منظمی و با استفاده از هر تخمینی از ارزش حالت‌های دیگر که در دسترس باشد، به روز می‌کنند. ارزش برخی حالت‌ها ممکن است قبل از اینکه ارزش دیگران یکبار به روز شود، چندین بار به روز شوند. با این حال، برای همگرایی صحیح، یک الگوریتم ناهمزمان باید بروزرسانی مقادیر همه حالت‌ها را ادامه دهد و نمی‌تواند حالتی را نادیده بگیرد. الگوریتم‌های ناهمزمان DP امکان انعطاف‌پذیری زیادی را در سیستم فراهم می‌کنند و در بسیاری از مسائل، تنها گزینه قابل اجرا به حساب می‌آیند.

۲-۵ روش‌های یادگیری تفاوت زمانی

برخلاف روش‌های برنامه‌ریزی پویا، روش‌های یادگیری تفاوت زمانی (TD) می‌توانند مستقیماً از تجربه خام بدون در اختیار داشتن مدلی از دینامیک محیط، یاد بگیرند. مانند DP، روش‌های TD ارزش‌ها حالت (حالت-عمل) را بر اساس تخمین‌های حالت (حالت-عمل)‌های دیگر به روز می‌کنند.

همانطور که دیدیم، ارزش یک حالت، امید ریاضی عایدی، با شروع از آن حالت است. بنابراین، یک روش آشکار برای تخمین ارزش حالت s از طریق تجربه، میانگین گرفتن تمام عایدی‌های تجربه شده در هر بار عبور از s است. با تجربه بیشتر و مشاهده عایدی‌های بیشتر، میانگین آن‌ها باید به $v_{\pi}(s)$ همگرا شود.

روش‌های تفاوت زمانی در ساده‌ترین حالت، بعد از عبور از هر حالت، ارزش آن حالت را با استفاده از ارزش حالت بعدی به عنوان تخمینی از عایدی، بلافاصله به روزرسانی می‌کنند

$$V(S_t) \leftarrow V(S_t) + \alpha [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

الگوریتم ۳ TD(۰) برای تخمین v_π ورودی: خط مشی π ۱: تابع $V(s)$ را با مقادیر دلخواه مقداردهی اولیه کن. برای هر گام در اپیزود۲: عمل a را با دنبال کردن خط مشی π انتخاب کن۳: پاداش r و حالت جدید s' را مشاهده کن۴: $V(s) \leftarrow V(s) + \alpha [r + \gamma V(s') - V(s)]$ ۵: $s \leftarrow s'$

۶:

۱-۵-۲ Q-learning

ایده اصلی در روش Q-learning، تخمین تابع مقدار عمل $Q^*(s, a)$ با استفاده از معادله بلمن به عنوان یک بروزرسانی تکراری،

$$Q_{i+1}(s, a) = \mathbb{E}[r + \gamma \max_{a'} Q_i(s', a') | s, a]$$

چنین الگوریتم‌های تکرار مقداری به تابع ارزش عمل بهینه همگرا می‌شوند، $Q_i \rightarrow Q^*$ وقتی $i \rightarrow \infty$ ، این رویکرد کلی کاملاً غیر عملی است. زیرا تابع ارزش عمل برای هر دنباله، به طور جداگانه و بدون هیچ گونه تعمیم برآورد می‌شود. در عوض، معمولاً از یک تخمین‌گر توابع (مثل شبکه عصبی) برای تخمین تابع ارزش عمل استفاده می‌شود. در فصل سوم با این روش بیشتر آشنا خواهیم شد.

فصل ۳

نتایج اخیر

۳-۱ روش‌های مبتنی بر مدل و بدون مدل

یکی از مهم‌ترین نقاط انشعاب در الگوریتم‌های RL این است که آیا عامل به یک مدل از محیط دسترسی دارد یا توانایی آموختن مدلی از محیط را دارد؟ منظور از مدل محیط، تابعی است که انتقال و پاداش هر حالت-عمل را پیش‌بینی می‌کند. نقطه قوت داشتن مدل این است که به عامل اجازه می‌دهد با برآورد قبلی، طیف وسیعی از گزینه‌های ممکن را پیش‌بینی کند و به صراحت در مورد گزینه‌های خود تصمیم بگیرد. سپس عامل می‌تواند نتایج حاصل از برنامه‌ریزی قبلی را در قالب یک خط‌مشی بیاموزد یک نمونه مشهور از این روش AlphaZero است. در عمل، چنانچه دستیابی به مدلی از محیط امکان‌پذیر و عملی باشد، معمولاً از روش‌های مبتنی بر مدل استفاده می‌شود. زیرا می‌تواند باعث بهبود قابل توجه‌ای در کارایی نمونه نسبت به روش‌های بدون مدل شود. اصلی‌ترین نقطه ضعف این روش‌ها این است که معمولاً یک مدل کامل از محیط در دسترس عامل نیست و مدل کاملاً از طریق تجربه یاد گرفته می‌شود. الگوریتم‌هایی که از یک مدل استفاده می‌کنند، روش‌های مبتنی بر مدل و آن‌هایی که از چنین مدلی استفاده نمی‌کنند، بدون مدل نامیده می‌شوند. روش‌های بدون مدل از دستاوردهای بالقوه روش‌های مبتنی بر مدل چشم‌پوشی می‌کنند، اما پیاده‌سازی و تنظیم آنها آسان‌تر است. به همین خاطر، روش‌های بدون مدل از محبوبیت بیشتری برخوردار بوده و به طور گسترده‌تری توسعه و آزمایش شده‌اند.

۲-۳ روش‌های بدون مدل

۱-۲-۳ روش‌های مبتنی بر ارزش

در روش‌های یادگیری تقویتی بدون مدل مبتنی بر ارزش تابع ارزش عمل با استفاده از یک تخمین‌گیر تابع^۱، مانند شبکه عصبی، نشان داده می‌شود. فرض کنید $Q(s, a; \theta)$ یک تابع ارزش عمل تقریبی با پارامتر θ باشد الگوریتم‌های مختلفی برای بروزرسانی θ وجود دارد الگوریتم $Q-learning$ یکی از نمونه‌های چنین الگوریتمی است که هدف آن تقریب مستقیم تابع ارزش عمل بهینه $Q^*(s, a) \approx Q(s, a; \theta)$ است. در $Q-learning$ یک مرحله‌ای، پارامترهای θ از تابع ارزش عمل با به حداقل رساندن تابع هزینه به شکل مرحله به مرحله آموخته می‌شوند، به شکلی که تابع هزینه i ام به شکل

$$L_i(\theta_i) = \mathbb{E} \left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right)^2$$

تعریف می‌شود که s' حالتی است که بعد از حالت s دیده می‌شود.

روش‌های Q-learning

خانواده روش‌های Q-learning تلاش می‌کنند مستقیماً تابع ارزش عمل-حالت بهینه $Q^*(s, a)$ را تخمین بزنند. آن‌ها به طور معمول از یک تابع هدف مبتنی بر معادله بلمن استفاده می‌کنند. این بهینه‌سازی تقریباً همیشه به صورت off-policy انجام می‌شود، به این معنی که هر بروزرسانی می‌تواند از داده‌های جمع‌آوری شده در هر نقطه استفاده کند، بدون اینکه در نظر بگیرد نحوه انتخاب عامل برای کشف محیط در هنگام بدست آوردن داده‌ها چگونه بوده است. خط‌مشی مربوطه از طریق ارتباط بین Q^* و π^* بدست می‌آید:

عامل بعد از یادگرفتن تابع $Q_\theta(s, a)$ به طوری که $Q_\theta(s, a) \approx Q^*(s, a)$ می‌تواند عمل بهینه در حالت s را به صورت زیر محاسبه کند

$$a(s) = \arg \max_a Q_\theta(s, a).$$

از جمله الگوریتم‌های Q-learning می‌توان به موارد زیر اشاره کرد:

¹Function approximator

- روش کلاسیک DQN که حوزه یادگیری تقویتی ژرف^۲ را عمیقاً ارتقا بخشید.
- روش C5۱ که توزیعی روی عایدی را می‌آموزد که امیدریاضی آن Q^* است.

روش DQN

معمولاً برای تقریب زدن توابع ارزش در یادگیری تقویتی، از یک تابع خطی استفاده می‌شود. اما گاهی اوقات از یک تقریب عملکرد غیرخطی به جای آن، مانند یک شبکه عصبی استفاده می‌شود. شبکه‌های عصبی با عنوان شبکه Q^3 شناخته می‌شوند. شبکه Q را می‌توان با کمینه ساختن دنباله‌ای از توابع هزینه به شکل $L_1(\theta_1), L_2(\theta_2), L_3(\theta_3), \dots$ آموزش داد؛ به طوری‌که

$$L_i(\theta_i) = \mathbb{E} \left[(y_i - Q(s, a; \theta_i))^2 \right]$$

و

$$y_i = \mathbb{E}[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) | s, a].$$

با مشتق گرفتن از تابع هزینه نسبت به پارامترهای θ_i خواهیم داشت:

$$\nabla_{\theta_i} L_i \theta_i = \mathbb{E} \left[(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i)) \nabla_{\theta_i} Q(s, a; \theta_i) \right].$$

به جای محاسبه امیدریاضی کامل در گرادیان فوق، غالباً از نظر محاسباتی، بهینه‌سازی تابع هزینه با نزول گرادیان تصادفی^۴ راه‌حل بهتری است. اگر در هر مقطع زمانی، وزن‌ها بروزرسانی شود و امیدریاضی با یک نمونه از توزیع خط‌مشی رفتار^۵ جایگزین شود، الگوریتم Q -learning^{۱۲} را نشان می‌دهد.

توجه داشته باشید که ۱۲ یک الگوریتم بدون مدل است: این کار وظیفه یادگیری تقویتی را مستقیماً با استفاده از نمونه‌های شبیه ساز E بدون ساختن صریح تخمین E حل می‌کند. استراتژی حریصانه، $a = \max_a Q(s, a; \theta)$ ، در حالی که یاد می‌گیرد که کاوش کافی در فضای حالت را تضمین کند.!!!!!!!!!!!!!! . توزیع رفتار اغلب توسط یک استراتژی Greedy انتخاب می‌شود که استراتژی حریصانه را با احتمال ۱ دنبال می‌کند و یک اقدام تصادفی با احتمال ϵ !!!!!!!!!!!!!!!

²Deep reinforcement learning

³Q-Network

⁴stochastic gradient descend

⁵behavior policy

الگوریتم ۴ الگوریتم Q-learning با replay Experience

- ۱: حافظه D replay را مقدار دهی اولیه کن
 - ۲: تابع ارزش عمل Q را با وزنهای تصادفی مقداردهی اولیه کن
 - ۳: برای هر اپیزود $1 \dots M$:
 - ۴: دنباله $d_1 = \{S_1\}$ و کدینگ $\phi_1 = \phi(d_1)$ را مقداردهی اولیه کن
 - ۵: برای $t = 1 \dots T$:
 - ۶: با احتمال ϵ یک عمل تصادفی a_t را انتخاب کن، در غیر این صورت $a_t = \max_a Q^*(\phi(d_t), a; \theta)$ را انتخاب کن
 - ۷: عمل a_t را انجام بده و حالت S_{t+1} و پاداش R_t را مشاهده کن
 - ۸: قرار بده $d_{t+1} = d_t, a_t, S_{t+1}$ و $\phi_{t+1} = \phi(d_{t+1})$
 - ۹: تجربه $(\phi_t, A_t, R_t, \phi_{t+1})$ را در D ذخیره کن
 - ۱۰: یک نمونه تصادفی از تجربه‌های $(\phi(j), A_j, R_j, \phi_{j+1})$ از انبار تجربیات D انتخاب کن
 - ۱۱: قرار بده
$$y_j = \begin{cases} r_j & \phi_{j+1} \text{ terminal} \\ r_j & \text{otherwise} \end{cases}$$
 - ۱۲: یک گام از نزول گرادیان را برای تابع هزینه $(y_j - Q(\phi_j, a_j; \theta))^2$ انجام بده
-

روش C۵۱

روش DDPG

گسسته سازی فضای عمل یک روش برای سازگار کردن و انطباق روش‌های یادگیری تقویتی عمیق نظیر DQN با دامنه‌های پیوسته می‌باشد. با این حال، این روش محدودیت‌های زیادی دارد، مخصوصاً مشکل نفرین ابعاد. نفرین ابعاد بیانگر این است که تعداد عمل‌ها به صورت نمایی با تعداد درجات آزادی افزایش پیدا می‌کند.

در روشی جدید یک الگوریتم بازیگر-منتقد مستقل از خط‌مشی، مستقل از مدل^۶ با استفاده از تخمین‌گر تابع عمیق ارائه می‌دهند که می‌تواند خط‌مشی‌ها را در فضاهاى عمل پیوسته با ابعاد بالا یاد بگیرد.

این روش بر اساس الگوریتم گرادیان خط‌مشی معین^۷ است که آن را گرادیان خط‌مشی معین عمیق^۸ می‌نامیم. در آن روش، رویکرد بازیگر-منتقد را با بینش شبکه Q عمیق^۹ ترکیب می‌کنیم.

الگوریتم DPG تابع عمل پارامتری $\mu(s|\theta^\mu)$ را نگهداری می‌کند که با نگاشت معین حالت‌ها به یک عمل خاص، سیاست فعلی را مشخص می‌کند.

همانند روش یادگیری Q در اینجا نیز منتقد $Q(s, a)$ با استفاده از معادله بلمن آموخته می‌شود. بازیگر با پیروی از اعمال قاعده زنجیره‌ای روی عایدی چشم‌داشتی از توزیع شروع J نسبت به پارامترهای بازیگر به روز می‌شود.

$$\begin{aligned} \nabla_{\theta^\mu} J &= \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t|\theta^\mu)}] \\ &= \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s=s_t}] \end{aligned} \quad (۱-۳)$$

در این روش نیز، مانند روش DQN از انبار تجربه استفاده می‌شود. در هر مرحله، بازیگر و منتقد با نمونه‌برداری یکنواخت از انبار به روزرسانی می‌شوند. از آنجا که DDPG الگوریتمی مستقل از خط‌مشی می‌باشد، انبار تجربه می‌تواند بزرگ باشد، که به الگوریتم اجازه می‌دهد تا از یادگیری مجموعه‌ای از انتقال‌های ناهمبسته بهره‌مند شود. چالش اصلی یادگیری در فضاهاى عمل پیوسته، اکتشاف است. یک

^۶ A model-free, off-policy actor-critic algorithm

^۷ Deterministic policy gradient (DPG)

^۸ Deep DPG (DDPG)

^۹ Deep Q Network (DQN)

مزیت الگوریتم‌های مستقل از خط‌مشی مانند DDPG این است که می‌توان به مسئله کاوش، مستقل از الگوریتم یادگیری پرداخت. می‌توان یک خط‌مشی اکتشاف μ' را با اضافه کردن نویز (نمونه‌گیری شده از فرآیند \mathcal{N}) به خط‌مشی بازیگر ساخت:

$$\mu'(s_t) = \mu(s_t | \theta_t^m u) + \mathcal{N} \quad (2-3)$$

می‌توان \mathcal{N} را متناسب با محیط انتخاب کرد.

۲-۲-۳ روش‌های مبتنی بر خط‌مشی

در این قسمت روش‌هایی را در نظر می‌گیریم که برای دستیابی به خط‌مشی بهینه، به جای استفاده از تابع ارزش عمل یا ارزش حالت، یک خط‌مشی پارامتری شده^{۱۰} را می‌آموزد. با این حال ممکن است برای یادگیری پارامترهای خط‌مشی از یک تابع ارزش استفاده شود، اما تابع ارزش برای انتخاب عمل مورد نیاز نیست. ما از نماد $\theta \in \mathbb{R}^d$ برای بردار پارامتر خط‌مشی استفاده می‌کنیم. برخلاف روش‌های مبتنی بر ارزش، روش‌های مبتنی بر خط‌مشی مستقیماً تابع خط‌مشی $\pi_\theta(a|s)$ را تخمین می‌زنند و پارامترهای θ را با استفاده از صعود گرادیان^{۱۱} روی یک مقیاس عملکرد^{۱۲} $J(\pi_\theta)$ یا به طور مستقیم و یا با بیشینه‌سازی تخمین‌های محلی از $J(\pi_\theta)$ بروزرسانی می‌کنند. این روش تقریباً همیشه به صورت on-policy عمل می‌کند.

همانطور که در ادامه خواهیم دید، می‌توان از توابع مختلفی برای مقیاس عملکرد J استفاده نمود. یک انتخاب بدیهی $J(\pi_\theta) = \mathbb{E}[R_t]$ است. هدف این روش‌ها بیشینه کردن تابع J است. در عبارت

$$\theta_{t+1} = \theta_t + \alpha \nabla \hat{J}(\theta_t)$$

$\nabla \hat{J}(\theta_t)$ ، تخمینی احتمالاتی است که امید ریاضی آن گرادیان J را نسبت به پ θ_t تخمین می‌زند.

به روش‌هایی که چنین الگویی را برای محاسبه خط‌مشی بهینه دنبال می‌کنند، روش‌های گرادیان خط‌مشی^{۱۳} می‌گوییم. دسته‌ای از روش‌های گرادیان خط‌مشی وجود دارند که تلاش می‌کنند تخمینی از

¹⁰Parameterized

¹¹Gradient ascent

¹²Performance Measure

¹³Policy Gradient

الگوریتم ۵ الگوریتم DDPG

۱: پارامترهای θ_μ و θ_Q به ترتیب مربوط به بازیگر $\mu(s; \theta_\mu)$ و منتقد $Q(s, a; \theta_Q)$ را مقداردهی اولیه کن.

۲: پارامترهای توابع هدف μ' و Q' را با وزنهای $\theta_\mu \longleftrightarrow \theta_{\mu'}$ و $\theta_Q \longleftarrow \theta_{Q'}$ مقداردهی اولیه کن

۳: حافظه تکرارها R را بساز برای هر اپیزود $1 \dots M$

۴: یک تابع نویز تصادفی \mathbb{N} بساز

۵: حالت اولیه S_1 را مشاهده کن برای $t = 1 \dots T$

۶: عمل $a_t = \mu(s_t; \theta_\mu) + \mathbb{N}_t$ را بر اساس خطمشی فعلی و نویز اکتشاف، انتخاب کن و حالت بعدی S_{t+1} و پاداش R_t را مشاهده کن.

۷: تجربه (s_t, a_t, r_t, s_{t+1}) را در انبار تجربه R ذخیره کن

۸: یک نمونه به اندازه N از تجربه‌های (s_i, a_i, r_i, s_{i+1}) از انبار تجربه R انتخاب کن

۹: وزنهای منتقد θ_Q را با در نظر گرفتن تابع هزینه $L = \frac{1}{N} \sum_i (y_i - Q(s_i, A_i; \theta_Q))^2$ بروزرسانی کن

۱۰: وزنهای بازیگر θ را با استفاده از گرادیان خطمشی نمونه

$$\nabla_{\theta_\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a; \theta_Q)|_{s=s_i, a=\mu(s_i)} \nabla_\mu \mu(s; \theta_\mu)|_{S_i}$$

به بروزرسانی کن

۱۱: وزنهای توابع هدف را به شکل

$$\theta_{Q'} = \tau \theta_Q + (1 - \tau) \theta_{Q'} \theta_{\mu'} = \tau \theta_\mu + (1 - \tau) \theta_{\mu'}$$

بروزرسانی کن

۱۳:

۱۴:

تابع ارزش را نیز محاسبه کنند. به چنین روش‌هایی، بازیگر-منتقد^{۱۴} گفته می‌شود که بازیگر^{۱۵} اشاره به خط‌مشی آموخته شده و منتقد^{۱۶} اشاره به تابع ارزش آموخته شده (معمولا یک تابع ارزش حالت) دارد.

قضیه‌ی ۳-۱ (گرادیان خط‌مشی)

$$\nabla J(\pi_\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s, a) \nabla_\theta \pi_\theta(a|s)$$

که μ یک توزیع احتمال روی S است که متناسب با تعداد دفعاتی است که حالت s با دنبال کردن خط‌مشی π_θ تکرار می‌شود.

می‌توان نشان داد

$$\nabla J(\pi_\theta) = \mathbb{E}_\pi \left[R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} \right]$$

بنابراین در هر گام $\left[R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} \right]$ یک تخمین‌گر ناریب از $\nabla J(\pi_\theta)$ خواهد بود [۱]. پس می‌توان در هر گام θ را به شکل زیر بروزرسانی کرد

$$\theta_{t+1} = \theta_t + \alpha R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} = \theta_t + \alpha R_t \nabla_\theta \log \pi_\theta(a|S_t)$$

چند نمونه از روش‌های بهینه‌سازی خط‌مشی به شرح زیر است.

روش‌های بازیگر منتقد^{۱۷} که الگوریتم گرادیان افزایشی را مستقیما برای بهینه‌سازی $J(\pi_\theta)$ به کار می‌برند.

روش Proximal Policy Optimization که!!!!!!!!!!!!

روش‌های بازیگر-منتقد

یک نمونه از روش‌های بهینه‌سازی خط‌مشی، خانواده REINFORCE از الگوریتم‌های یادگیری تقویتی است [۲]. الگوریتم استاندارد REINFORCE پارامترهای θ را در جهت $\nabla_\theta \log \pi(a_t|s_t : \theta) R_t$

¹⁴ Actor-Critic

¹⁵ Actor

¹⁶ Critic

¹⁷ Actor-Critic

بروزرسانی می‌کند که یک تخمین نااریب از $\nabla_{\theta} \mathbb{E}[R_t]$ است. می‌توان با تفریق یک تابع آموخته شده روی حالت‌ها، $b_t(s_t)$ از R_t ، واریانس این تخمین را کاهش داد بطوری‌که نااریب باقی بماند. به چنین تابعی پایه گفته می‌شود. در نتیجه گرادیان به شکل $\nabla_{\theta} \log \pi(a_t|s_t; \theta)(R_t - b_t(s_t))$ خواهد بود. معمولاً از تخمینی آموخته‌شده از تابع ارزش به عنوان پایه استفاده می‌شود، $b_t(s_t) \approx v_{\pi}(s_t)$ ، که منجر به تخمینی با واریانس بسیار کوچک‌تر از گرادیان خط‌مشی می‌شود؛ در حالیکه تخمین نااریب باقی می‌ماند و در نتیجه عملیات یادگیری با سرعت بیشتری انجام می‌شود. این روش می‌تواند به شکل معماری بازیگر-منتقد تعبیر شود که خط‌مشی π بازیگر و پایه b_t منتقد است. اگر از تخمین یک تابع ارزش حالت به عنوان پایه استفاده کنیم $b_t(s) = v_{\pi_{\theta_t}}(s)$ عبارت $R_t - b_t(s)$ می‌تواند به شکل تخمینی از مزیت^{۱۸} عمل a_t در حالت s_t یا $A(a_t, s_t) = Q(a_t, s_t) - v(s_t)$ تعبیر شود. چراکه R_t تخمینی از $Q_{\pi}(a_t, s_t)$ و b_t تخمینی از $v_{\pi}(s_t)$ است. در این صورت به این روش Advantage Actor-Critic یا A۲C گفته می‌شود.

الگوریتم ۶ الگوریتم Advantage Actor – Critic [۱]

ورودی: یک پارامتری‌شده مشتق‌پذیر از خط‌مشی $\pi_{\theta}(a|s)$

ورودی: یک پارامتری‌شده مشتق‌پذیر از تابع ارزش حالت $v_{\omega}(s)$

۱: پارامترهای خط‌مشی $\theta \in \mathbb{R}^{d'}$ و تابع ارزش حالت $\omega \in \mathbb{R}^d$ را مقداردهی اولیه کن

۲: تکرار کن:

۳: حالت اولیه S را بساز

۴: $I \rightarrow 1$

۵: تا وقتی S حالت نهایی نیست:

۶: $A \sim \pi_{\theta}(\cdot|S)$

۷: عمل A را انجام بده و حالت S' و پاداش R را مشاهده کن

۸: $R + \gamma v_{\omega}(S') - v_{\omega}(S) \rightarrow \delta$

۹: $\omega + \alpha^{\omega} I \delta \nabla_{\omega} v_{\omega}(S) \rightarrow \omega$

۱۰: $\theta + \alpha^{\theta} I \delta \nabla_{\theta} \ln \pi_{\theta}(A|S) \rightarrow \theta$

۱۱: $\gamma I \rightarrow I$

۱۲: $S' \rightarrow S$

¹⁸ Advantage

روش TRPO

تعریف ۳-۱ اگر $\rho_\pi : S \rightarrow \mathbb{R}$ تابع فرکانس تخفیف دار دیده شدن حالت ها باشد. یعنی

$$\rho_\pi(s) = P(S_0 = s) + \gamma P(S_1 = s) + \gamma^2 P(S_2 = s) + \dots$$

که دنباله S_0, S_1, S_2 خط مشی π را دنبال می کند.

اگر π و π' دو خط مشی باشند و $J(\pi) = \mathbb{E}_\pi[R_0]$ می توان نشان داد

$$J(\pi') = J(\pi) + \sum_s \rho_{\pi'}(s) \sum_a \pi'(a|s) A_\pi(s, a)$$

که $A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s)$ تابع مزیت عمل a در حالت s باشد. وابستگی پیچیده $\rho_{\pi'}(s)$ در طرف راست تساوی به π' بهینه سازی مستقیم را مشکل می کند. برای حل این مشکل شولمن و همکارانش مقیاس عملکرد دیگری، $L_\pi(\pi')$ ، را معرفی می کند و نشان می دهد که اگر π و π' به اندازه کافی به یکدیگر نزدیک باشند، افزایش $L_\pi(\pi')$ همواره با افزایش J همراه خواهد بود [۲].

$$L_\pi(\pi') = J(\pi) + \sum_s \rho_\pi(s) \sum_a \pi'(a|s) A_\pi(s, a) \quad (3-3)$$

در عبارت ۳-۳ از تابع فرکانس ρ_π به جای $\rho_{\pi'}$ استفاده می کند.

قضیه ۳-۲ فرض کنید $\alpha = D_{TV}^m ax(\pi_{old}, \pi_{new})$ باشد که

$$D_{TV}^m ax(\pi, \pi') = \max_s D_{TV}(\pi(\cdot|s) || \pi'(\cdot|s))$$

و $D_{TV}(p||q)$ دیورژانس *total variation* بین دو بردار p و q باشد

$$D_{TV}(p||q) = \frac{1}{2} \sum_i |p_i - q_i|$$

در این صورت

$$J(\pi_{new}) \geq L_{\pi_{old}}(\pi_{new}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}$$

که $\epsilon = \max_{s,a} |A_\pi(s, a)|$

با توجه به قضیه ۲-۳ و نامعادله $D_{TV}(p||q)^2 \leq D_{KL}(p||q)$ که $D_{KL}(p||q)$ برابر با دیورژانس KL دو بردار p و q است [۳]. می توان نتیجه گرفت

$$J(\pi') \geq L_{\pi}(\pi') - C D_{KL}^m ax(\pi, \pi')$$

که

$$C = \frac{4\epsilon\gamma}{(1-\gamma)^2} \quad (4-3)$$

رابطه ۲-۴ نشان می دهد که می توان یک دنباله صعودی از خط مشی ها داشت به طوری که

$$J(\pi_0) \leq J(\pi_1) \leq J(\pi_2) \leq \dots$$

چرا که فرض کنید $M_i(\pi) = L_{\pi_i}(\pi) - C D_{KL}^m ax(\pi_i, \pi)$ در این صورت

$$J(\pi_{i+1}) \geq M_i(\pi_{i+1})$$

$$J(\pi_i) = M_i(\pi_i)$$

بنابراین

$$J(\pi_{i+1}) - J(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i)$$

می توان نتیجه گرفت با بیشینه کردن M_i در هر گام می توان اطمینان حاصل کرد که مقیاس عملکرد واقعی J غیرنزولی خواهد بود.

اگر \hat{A}_t تخمین مزیت $A_{\pi_t}(S_t, A_t)$ باشد که در گام t محاسبه می شود، می توان نشان داد که در روش TRPO مقیاس عملکرد L_{π} در هر گام به شکل؛

$$\mathbb{E}_t \left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)} \hat{A}_t \right]$$

خواهد بود.

الگوریتم ۷ $PolicyIteration$ با مقیاس عملکرد L_π

- ۱: خط‌مشی π_0 را مقداردهی اولیه کن
 - ۲: برای $i = 0, 1, \dots$ تکرار کن:
 - ۳: همه مزیت‌های $A_{\pi_i}(s, a)$ را محاسبه کن
 - ۴: $\arg \max_{\pi} [L_{\pi_i}(\pi) - C D_{KL}^m ax(\pi_i, \pi)] \rightarrow \pi_{i+1} =$ که
- $$C = (\epsilon \gamma) / (1 - \gamma)^2$$

و

$$L_{\pi_i}(\pi) = J(\pi_i) + \sum_s \rho_{\pi_i}(s) \sum_a \pi(a|s) A_{\pi_i}(s, a)$$

روش PPO

در روش TRPO دیدیم که بیشینه‌سازی مقیاس عملکرد L_π ساده‌تر از مقیاس عملکرد J است ولی در عوض الگوریتم صعود گرادینان تنها مجاز به اعمال تغییرات کوچک در خط‌مشی است. یک راه دیگر برای کنترل تغییرات خط‌مشی استفاده از تابع CLIP است. در روش TRPO دیدیم که تابع مقیاس عملکرد، در گام t به شکل زیر است؛

$$L_{\pi_{old}}(\pi_{new}) = \mathbb{E}_{\approx} \left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)} A_\pi(S_t, A_t) \right]$$

فرض کنید $r_t(\theta_{new})$ نسبت احتمالات $\frac{\pi_{\theta_{new}}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ باشد. بنابراین

$$L_{\pi_{old}}(\pi_{new}) = \mathbb{E}_t \left[\frac{\pi_{\theta_{new}}(A_t|S_t)}{\pi_{\theta_{old}}(A_t|S_t)} A_\pi(S_t, A_t) \right] = \mathbb{E}_t [r_t(\theta_{new}) A_{\pi_{old}}(S_t, A_t)].$$

مقیاس عملکرد $L^{CLIP}(\theta)$ را به شکل زیر تعریف می‌کنیم

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[\min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t) \right]$$

که ϵ یک ابرپارامتر^{۱۹} مثلاً $\epsilon = 0.2$ است. اولین عبارت داخل \min همان مقیاس عملکرد روش TRPO است. در عبارت دوم مقادیر بزرگتر از $1 + \epsilon$ یا کوچکتر از $1 - \epsilon$ در $r_t(\theta)$ به ترتیب به $1 + \epsilon$

¹⁹Hyperparameter

و $\epsilon - 1$ تغییر پیدا کرده‌اند تا تغییرات بزرگ خط‌مشی را کنترل کنند. نهایتاً عبارت کوچکتر از میان این دو انتخاب خواهد شد. بنابراین اگر مقیاس عملکرد clip نشده (عبارت اول) کوچکتر یا مساوی با حالت clip شده (عبارت دوم) باشد، L^{CLIP} دقیقاً همان مقیاس عملکرد L_{π} خواهد بود. در غیر این صورت مقیاس عملکرد clip شده انتخاب می‌شود تا از تغییرات بزرگ خط‌مشی جلوگیری شود.

۳-۲-۳ مقایسه روش بهینه‌سازی خط‌مشی و Q-learning

نقطه قوت اصلی روش‌های بهینه‌سازی خط‌مشی، اصولی بودن آنهاست، به این معنا که شما مستقیماً چیزی که می‌خواهید را بهینه‌سازی می‌کنید. در نتیجه این روش‌ها قابل اتکا و باثبات هستند. در مقابل، روش‌های Q-learning با یادگیری تابع Q، مقیاس عملکرد را به طور غیر مستقیم بهینه می‌کند. حالت‌های زیادی برای این نوع یادگیری وجود دارد که به شکست منتهی می‌شود، بنابراین این روش‌ها ثبات کمتری دارند [۱]. روش‌های Q-learning می‌توانند از داده‌ها به طور موثرتری نسبت به تکنیک‌های بهینه‌سازی خط‌مشی استفاده کنند.

عامل بین بهینه‌سازی خط‌مشی و Q-learning بهینه‌سازی خط‌مشی و Q-learning ناسازگار نیستند (و به نظر می‌رسد تحت برخی شرایط، معادل آن باشد) و طیف وسیعی از الگوریتم‌ها وجود دارد که بین دو حد این طیف وجود دارند. الگوریتم‌هایی که در این طیف قرار دارند قادرند از نقاط قوت طرفین طیف استفاده کنند.

به طور مثال، DDPG الگوریتمی است که یک خط‌مشی قطعی و یک تابع Q را یاد می‌گیرد، به طوری که از هریک برای بهبود دیگری استفاده می‌کند. روش SAC، از خط‌مشی‌های تصادفی، تنظیم آنتروپی^{۲۰} و چند ترفند دیگر برای یادگیری و کسب امتیاز بالاتر از DDPG در محک‌های استاندارد^{۲۱} استفاده می‌کند.

²⁰entropy regularization

²¹Standard Benchmark

۳-۳ روش‌های مبتنی بر مدل

۱-۳-۳ روش مدل جهان

مدل جهانی^{۲۲} می‌تواند به سرعت و به روشی بدون نظارت آموزش ببیند تا یک بازنمایی از محیط را بیاموزد. سپس با استفاده از ویژگی‌های استخراج شده از مدل جهان به عنوان ورودی به یک عامل، می‌توان یک خط‌مشی ساده و فشرده را آموخت که می‌تواند وظیفه مورد نیاز را حل کند. حتی می‌توانیم عامل را کاملاً در داخل محیط رویایی خود که توسط مدل جهانی آن ایجاد شده، آموزش دهیم و این خط‌مشی آموخته شده را به محیط واقعی انتقال دهیم. در بسیاری از مسائل یادگیری تقویتی مبتنی بر مدل، عامل به مدل قدرتمندی از دینامیک محیط دسترسی دارد.

اکثر رویکردهای مبتنی بر مدل موجود در یادگیری تقویتی، مدلی از محیط را یاد می‌گیرند، اما همچنان در محیط واقعی آموزش می‌بینند. در این روش، ما همچنین می‌توانیم یک محیط مصنوعی را کاملاً جایگزین محیط واقعی کنیم و خط‌مشی عامل خود را فقط در داخل محیط مصنوعی آموزش دهیم و در نهایت خط‌مشی آموخته شده را به محیط واقعی انتقال دهیم.

²²World Models

پیوست آ

مطالب تکمیلی

پیوست‌های خود را در صورت وجود می‌توانید در این قسمت قرار دهید.

مراجع

- [1] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [2] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.
- [3] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

واژه‌نامه

الف

pallet پالت	heuristic ابتکاری
robustness پایداری	worth ارزش
support پشتیبان	satisfiability ارضاپذیری
convex hull پوسته‌ی محدب	strategy استراتژی
upper envelope پوش بالایی	coalition ائتلاف
covering پوششی	

ب

projective transformation تبدیل تصویری	loading بارگذاری
equilibrium تعادل	game بازی
relaxation تعدیل	label برچسب
intersection تقاطع	linear programming برنامه‌ریزی خطی
partition تقسیم‌بندی	integer programming برنامه‌ریزی صحیح
evolutionary تکاملی	packing بسته‌بندی
distributed توزیع‌شده	best response بهترین پاسخ
	maximum بیشینه

ج

brute-force جست‌وجوی جامع
Depth-First Search جست‌وجوی عمق‌اول

پ

س	bin جعبه
constructive ساختی	
pay off, utility سود	چ
	sink چاله
ش	
quasi-polynomial شبه‌چندجمله‌ای	ح
quasi-concave شبه‌مقعر	حرکت action
ص	خ
formal صوری	selfish خودخواهانه
	clique خوشه
ع	د
rational عاقل	دودویی binary
agent-based عامل-محور	dual دوگان
action عمل	bimatrix دو ماتریسی
غ	ر
missing غائب	vertex رأس
decentralized غیرمتمرکز	behaviour رفتار
degenerate غیرمعمول	coloring رنگ‌آمیزی
ق	ز
transferable قابل انتقال	scheduling زمان‌بندی
lexicographically قاموسی	biology زیست‌شناسی
strong قوی	
ک	

art gallery نگارخانه‌ی هنر	minimum کمینه
gaurd نگهبان	
profile نمایه	م
round-robin نوبتی	مجموع زیرمجموعه‌ها subset sum
	مجموعه set
و	محور pivot
facet وجه	مختلط mixed
	مخفی hidden
ه	مستوی affine
price of anarchy (POA) هزینه‌ی آشوب	مسطح planar
social cost هزینه‌ی اجتماعی	منطقی reasonable
price of stability (POS) هزینه‌ی پایداری	موازی parallel
	ن
ی	نتیجه‌ی نهایی outcome
edge یال	نش Nash
isomorphism یکریختی	نقطه‌ثابت fixed point

Abstract

We present a standard template for typesetting theses in Persian. The template is based on the X_YPersian package for the L^AT_EX typesetting system. This write-up shows a sample usage of this template.

Keywords: Thesis, Typesetting, Template, X_YPersian



Sharif University of Technology

Department of Computer Engineering

M.Sc. Thesis

A Standard Template for Typesetting Theses in Persian

By:

Hamid Zarrabi-Zadeh

Supervisor:

Dr. Supervisor

September 2017