



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

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 بهره‌برداری
۱۰	۹-۱ اهداف تحقیق
۱۰	۱۰-۱ ساختار پایان‌نامه
۱۱	۲ مفاهیم اولیه یادگیری تقویتی
۱۲	۱-۲ فرایند تصمیم‌گیری مارکوف
۱۳	۲-۲ خط‌مشی
۱۴	۳-۲ سیگنال پاداش

۴.۲ section. ۱۴

۱۵	۵-۲ محیط
۱۶	۶-۲ خط‌مشی و تابع ارزش بهینه
۱۶	۷-۲ معادله بلمن
۱۷	۸-۲ بهینگی و معادله بهینگی بلمن
۱۸	۹-۲ برنامه‌ریزی پویا
۱۹	۱۰-۲ بهبود خط‌مشی
۲۰	۱۱-۲ الگوریتم خط‌مشی تکراری
۲۰	۱۲-۲ الگوریتم Iteration Value
۲۱	۱۳-۲ درهم تنیدگی PE و PI و الگوریتم های GPI
۲۱	۱۴-۲ Q-learning

۳ نتایج جدید

۲۳	۱-۳ روش های مبتنی بر مدل و بدون مدل
۲۴	۲-۳ روش های بدون مدل
۲۴	۳-۳ روش های مبتنی بر ارزش
۲۴	۴-۳ روش های مبتنی بر خط مشی
۲۶	۵-۳ روش های Actor-Critic
۲۸	۶-۳ روش TRPO
۳۰	۷-۳ روش PPO
۳۱	۸-۳ روش های Q-learning
۳۱	۹-۳ روش DQN
۳۴	۱۰-۳ روش C5۱

۳-۱۱ مقایسه روش بهینه سازی خط مشی و Q-learning ۳۴

۳-۱۲ روش DDPG ۳۴

۳-۱۳ روش SAC ۳۶

۳-۱۴ روش های مبتنی بر مدل ۳۶

آ مطالب تکمیلی ۳۸

فصل ۱

مقدمه

یک روش یادگیری ماشین تعریف می شود 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 صریحا در تعامل با یک محیط نامشخص بررسی می کند. به عامل هوشمند گفته نمی شود 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، می تواند بر وضعیت آینده محیط تأثیر بگذارد یادگیری تقویتی یک رویکرد محاسباتی برای درک و خودکار کردن یادگیری و تصمیم گیری هدفمند است. یادگیری تقویتی، یادگیری از طریق تعامل است که چگونه می توان برای رسیدن به یک هدف رفتار کرد. عامل یادگیری تقویتی و محیط، در طی مراحل زمانی گسسته یا پیوسته با یکدیگر تعامل دارند.

^۱ 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 می شود

تعریف دقیق تر این مسئله را در فصل دوم خواهیم دید.

۲-۱ اهمیت موضوع

یادگیری تقویتی در بسیاری از رشته ها مانند نظریه بازی، نظریه کنترل، تحقیق در عملیات، نظریه اطلاعات، بهینه سازی مبتنی بر شبیه سازی، سیستم های چند عاملی، هوش انبوه و آمار مورد مطالعه قرار می گیرد. در ادبیات تحقیق و کنترل عملیات، یادگیری تقویتی را برنامه ریزی تقریبی پویا^۲ یا برنامه ریزی عصبی پویا^۳ می نامند. 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 حل های بهینه و الگوریتم های 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 حالت ها و عمل ها

۴-۱ خط مشی

۵-۱ سیگنال پاداش

In reinforcement learning, the agent of the goal or purpose is formalized in terms of a special signal called the reward, which is passed from the environment to the agent.

^۲ Approximate Dynamic Programming

^۳ Neuro-dynamic Programming

^۴ Optimal Control Theory

agent's the Informally, R_t number, simple a is reward the step, time each At
 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

اصطلاح 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،

^۱Optimal control

^۲Dynamical system

^۳Richard Bellman

^۴Hamilton

^۵Jacobi

^۶Value function

^۷Bellman equation

^۸Markov decision process(MDP)

^۹Reinforcement learning (RL)

یعنی خط‌مشی^{۱۰}، سیگنال پاداش^{۱۱}، تابع ارزش و محیط^{۱۲} را دقیقاً تعریف خواهیم کرد. همچنین برخی روش‌های کلاسیک در یادگیری تقویتی را معرفی می‌کنیم.

۲-۱ فرایند تصمیم‌گیری مارکوف

یادگیری تقویتی از چارچوب رسمی فرایندهای تصمیم‌گیری مارکوف (MDP) برای تعریف تعامل بین یک عامل یادگیری و محیط آن 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 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 محدود است. بیشتر نظریه‌های فعلی یادگیری تقویتی، محدود به MDP متناهی است، اما روش‌ها و ایده‌ها به طور کلی بیان می‌شوند.

تعریف ۲-۱ (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^{۱۳} $\mathcal{P} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S} \times \mathcal{R})$ تابعی است که دینامیک MDP را مشخص می‌کند.

^{۱۰} Policy

^{۱۱} Reward signal

^{۱۲} Environment

^{۱۳} Probability transition kernel

هسته احتمال انتقال یا تابع انتقال P ، هر دوتایی حالت-عمل (s, a) ، که $s \in S$ و $a \in A$ ، را به یک توزیع احتمال روی دوتایی‌هایی به شکل (s', r) نسبت می‌دهد که s' بیانگر حالت بعدی و r بیانگر پاداش این انتقال است. به ازای هر دو حالت $s, s' \in S$ و هر عمل $a \in A$ 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 عضو $[0, 1]$ 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 هستند، بدین معنا که تابع انتقال تنها به حالت فعلی سیستم و آخرین عمل وابسته است، و نسبت به حالت‌ها و اعمال قبل از آن مستقل است.

۲-۲ خط‌مشی

خط‌مشی^{۱۴} نحوه رفتار عامل یادگیری را در یک زمان خاص، مشخص می‌کند و هسته اصلی رفتار یک عامل یادگیری تقویتی است. خط‌مشی به تنهایی برای تعیین رفتار کافی است. به عنوان یک تعریف غیر دقیق، خط‌مشی، نگاشتی از حالت‌های مدل شده از محیط به اقداماتی است که باید در آن حالت انجام شود. خط‌مشی ممکن است یک عملکرد ساده یا جدول جستجو باشد، یا ممکن است شامل محاسبات پیچیده‌ای مانند فرآیند جستجو باشد؛ همچنین خط‌مشی‌ها ممکن است تصادفی باشند.

تعریف ۲-۲ (خط‌مشی احتمالاتی ثابت) یک خط‌مشی احتمالاتی ثابت^{۱۵} (یا به طور خلاصه خط‌مشی ثابت) $\pi : S \rightarrow \Pi(A)$ حالت‌ها را به توزیع احتمال روی فضای عمل می‌نگارد. به طور خلاصه احتمال انتخاب عمل a از حالت s را با $\pi(a | s)$ نشان می‌دهیم.

می‌گوییم خط‌مشی π در یک MDP دنبال می‌شود هرگاه

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

^{۱۴} Policy

^{۱۵} Stationary probabilistic policy

۲-۳ سیگنال پاداش

یک سیگنال پاداش هدف را در یک مسئله یادگیری تقویتی تعریف می‌کند. در هر گام، محیط یک عدد حقیقی به نام پاداش برای عامل یادگیری تقویتی ارسال می‌کند. تنها هدف عامل، به حداکثر رساندن کل پاداش دریافتی در طولانی زمان است. سیگنال پاداش اتفاقات خوب و بد برای عامل را مشخص می‌کند و مبنای اصلی تغییر خط‌مشی است. سیگنال پاداش می‌تواند تابعی تصادفی از وضعیت محیط و اقدام انجام شده باشد.

۲-۴ عایدی^{۱۶} و تابع ارزش

سیگنال پاداش نشان می‌دهد که انجام چه عملی در هر گام خوب است، در حالی که تابع ارزش مشخص می‌کند که کدام خط‌مشی در طولانی مدت بهتر است. در واقع تابع ارزش نشانگر مطلوبیت طولانی مدت حالت‌ها پس از در نظر گرفتن حالت‌هایی است که احتمالاً در پی خواهند داشت. ارزش حالت s ، مجموع میزان پاداشی است که عامل می‌تواند انتظار داشته باشد با شروع از s 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 بیشتر 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، تخمین زده و مجدد برآورد شوند. مهمترین مولفه بیشتر الگوریتم‌های یادگیری تقویتی که در نظر می‌گیریم، روشی برای تخمین کارآمد تابع ارزش است.

تعریف ۲-۳ عایدی تخفیف دار آینده^{۱۷} یا به اختصار، عایدی، 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 $T = \infty$

^{۱۶} Return

^{۱۷} Future discounted Return

از تعریف بالا نتیجه می‌شود

$$\begin{aligned} G_t &= R_t + \sum_{t'=t+1}^T \gamma^{t'-t} R_{t'} \\ &= R_t + \gamma \sum_{t'=t+1}^T \gamma^{t'-(t+1)} R_{t'} \\ &= R_t + \gamma G_{t+1}. \end{aligned} \quad (۱-۲)$$

تعریف ۲-۴ (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.

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

تابع v_π را تابع ارزش حالت^{۱۸} مربوط به خط‌مشی π می‌نامیم.

تعریف ۲-۵ (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^{۱۹} مربوط به خط‌مشی π می‌نامیم.

۲-۵ محیط

۲-۵. محیط

مدل محیط رفتار محیط را تقلید می‌کند، یا به طور کلی‌تر، اجازه می‌دهد تا در مورد نحوه رفتار محیط، پیشبینی کارآمدی داشته باشیم. 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 نامیده می‌شوند. این روش‌ها در مقابل روش‌های بدون مدل هستند که همگی آزمون و خطا هستند.

^{۱۸} State Value Function
^{۱۹} Action Value Function

۶-۲ خطمشی و تابع ارزش بهینه

۶-۲. خطمشی و تابع ارزش بهینه

برای MDP های متناهی، می‌توانیم خطمشی بهینه^{۲۰} را به صورت زیر تعریف کنیم

تعریف ۶-۲ می‌گوییم خطمشی π بهتر یا مساوی خطمشی π' است و می‌نویسیم $\pi \geq \pi'$ هرگاه برای هر $s \in S$

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

می‌توان نشان داد که 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 ولی تابع ارزش متناظر با همه خطمشی‌های بهینه یکسان است و برابر با تابع ارزش بهینه است که با نماد v_* نمایش داده شده و به شکل زیر تعریف می‌شود. برای هر $s \in S$

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

همچنین تمام خطمشی‌های بهینه تابع عمل-ارزش مشترکی دارند که آن را با نماد q_* نمایش می‌دهیم و به شکل زیر تعریف می‌شود. برای هر دوتایی حالت عمل (s, a) که $s \in S$ و $a \in A(s)$

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

می‌توانیم q_* را برحسب v_* به شکل زیر بنویسیم

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \quad (۲-۲)$$

۷-۲ معادله بلمن

۷-۲. معادله بلمن

ویژگی اساسی توابع ارزش که در طول یادگیری تقویتی و برنامه‌ریزی پویا استفاده می‌شوند صدق کردن در روابط بازگشتی است. معادلات بلمن رابطه‌ای بین ارزش یک حالت و ارزش‌های حالت‌های بعدی

^{۲۰}Optimal policy

آن را بیان می‌کند

$$\begin{aligned}
 v_\pi(s) &\triangleq \mathbb{E}_\pi[G_t | S_t = s] \\
 &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s] \\
 &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s']] \\
 &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) [r + \gamma v_\pi(s')] \quad \forall s \in (S) \quad (۳-۲)
 \end{aligned}$$

۸-۲ بهینگی و معادله بهینگی بلمن

۸-۲. بهینگی و معادله بهینگی بلمن

$$\begin{aligned}
 v_*(s) &= \max_{a \in \mathbb{A}(s)} q_{\pi_*}(s, a) \\
 &= \max_a \mathbb{E}_{\pi_*}[G_t | S_t = s, A_t = a] \\
 &= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\
 &= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \\
 &= \max_a \sum_{s', r} p(s', r|s, a) [r + \gamma v_*(s')]. \quad (۴-۲)
 \end{aligned}$$

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

۹-۲ برنامه‌ریزی پویا

۲-۹. برنامه‌ریزی پویا

برنامه‌ریزی پویا^{۲۱} مجموعه‌ای از الگوریتم‌ها است که برای محاسبه خط‌مشی بهینه استفاده می‌شود. 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.

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

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')] \quad (۵-۲)$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')] \quad (۶-۲)$$

اگر دینامیک محیط کاملاً مشخص باشد معادلات ۵-۲ و ۶-۲ به ترتیب یک دستگاه معادلات خطی با $|S|$ است که راه‌حل آن سراسر است. برای اهداف ما، روش‌های تکراری مناسب‌ترین روش‌ها هستند. یک دنباله از توابع ارزش تقریبی v_0, v_1, v_2, \dots 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 موفق به عنوان یک قانون به‌روز رسانی به وسیله معادله بلمن برای v_π بدست می‌آید.

$$\begin{aligned} v_{k+1}(s) &\triangleq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')] \end{aligned} \quad (۷-۲)$$

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 به‌روز رسانی 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 تکراری^{۲۲} می‌نامند.

۲-۱۰. بهبود خط‌مشی

^{۲۱} Dynamic Programming (DP)^{۲۲} Iterative policy evaluation

۲-۱۰. بهبود خط‌مشی

دلیل ما برای محاسبه تابع ارزش یک خط‌مشی کمک به یافتن خط‌مشی‌های بهتر است. فرض کنید ما تابع ارزش v_π را برای یک خط‌مشی تعیین‌گرایانه دلخواه π تعیین کرده‌ایم. برای یک حالت s ما می‌خواهیم بدانیم که آیا باید خط‌مشی را برای انتخاب قطعی یک عمل a تغییر دهیم یا خیر. ما می‌دانیم که پیروی از خط‌مشی فعلی از حالت s چقدر خوب است، اما آیا تغییر به سیاست جدید بهتر است یا بدتر؟ یکی از راه‌های پاسخ به این سوال در نظر گرفتن انتخاب عمل a در حالت s و پس از آن پیروی از خط‌مشی موجود (π) است. ارزش این شیوه رفتار این است که ؟؟؟؟؟

قضیه‌ی ۲-۱ (قضیه‌ی بهبود خط‌مشی) فرض کنید π و π' دو خط‌مشی معین باشند که برای هر $s \in S$

$$q_\pi(s, \pi'(s)) \geq \pi'(s)$$

در این صورت $v_{\pi'}(s) \geq v_\pi(s)$

درستی قضیه بالا را می‌توان با استفاده از تعاریف به روشنی بررسی کرد

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

۲-۱۱. الگوریتم خط‌مشی تکراری

۲-۱۱. الگوریتم خط‌مشی تکراری

در الگوریتم خط‌مشی تکراری^{۲۳} هنگامی که یک خط‌مشی π با استفاده از v_π بهبود یافته است تا به خط‌مشی بهتری π_0 برسیم، سپس می‌توانیم v_π را محاسبه کرده و مجدداً آن را بهبود بخشیم تا π_{00} بهتر داشته باشیم. بنابراین ما می‌توانیم دنباله‌ای از خط‌مشی‌ها و تابع ارزش‌هایی بدست آوریم که به صورت یکنوا در حال بهبود هستند:

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

۲-۱۲. الگوریتم Iteration Value

۲-۱۲. الگوریتم ITERATION VALUE

یک اشکال در تکرار خط‌مشی این است که هر یک از تکرارهای آن شامل ارزیابی سیاست است که ممکن است باشد به خودی خود یک محاسبه تکراری طولانی مدت است که نیاز به رفت و برگشت‌های متعدد از طریق مجموعه حالت دارد. اگر سیاست ارزیابی بصورت تکراری انجام می‌شود، سپس همگرایی دقیقاً به v_{π} فقط در حد مجاز اتفاق می‌افتد. باید منتظر بمانیم برای همگرایی دقیق، یا می‌توانیم کوتاه‌تر از آن متوقف شویم؟ مثالی که در شکل ۱۰۴ وجود دارد قطعاً این را نشان می‌دهد کوتاه کردن ارزیابی سیاست ممکن است باشد. در آن مثال، تکرار ارزیابی سیاست فراتر از آن است سه مورد اول هیچ تاثیری در سیاست حریص مربوطه ندارند. در واقع، مرحله ارزیابی سیاست تکرار سیاست می‌تواند از چند طریق بدون ضرر کوتاه شود همگرایی تضمین سیاست را تضمین می‌کند. یک مورد خاص مهم، ارزیابی سیاسی است فقط پس از یک بار جابجایی متوقف می‌شود (یک به روزرسانی از هر حالت). این الگوریتم تکرار مقدار نامیده می‌شود. آی تی را می‌توان به عنوان یک عملیات به روزرسانی ساده نوشت که ترکیبی از بهبود سیاست و مراحل ارزیابی سیاست کوتاه شده: *policy to drawback One* itself may which evaluation, policy involves iterations its of each that is iteration state the through sweeps multiple requiring computation iterative protracted a be occurs v_{π} to exactly convergence then iteratively, done is evaluation policy If set.

of short stop we can or convergence, exact for wait we Must limit. the in only to possible be may it that suggests certainly ۱.۴ Figure in example The that? beyond iterations evaluation policy example, that In evaluation. policy truncate the fact, In policy. greedy corresponding the on effect no have three first the without ways several in truncated be can iteration policy of step evaluation policy case special important One iteration. policy of guarantees convergence the losing each of update (one sweep one just after stopped is evaluation policy when is particularly a as written be can It iteration. value called is algorithm This state). truncated and improvement policy the combines that operation update simple steps: evaluation policy

to converge to shown be can v_k sequence the v_0 , arbitrary For S . $\forall s$ all for $v[s]$. of existence the guarantee that conditions same the under $v[s]$

۱۳-۲ درهم تنیدگی PE و PI و الگوریتم های GPI

۱۳-۲. درهم تنیدگی PE و PI و الگوریتم های GPI

in- they that is far so discussed have we that methods DP the to drawback major A sweeps require they is, that MDP, the of set state entire the over operations involve pro- be can sweep single a even then large, very is set state the If set. state the of expensive hibitively

not are that algorithms DP iterative in-place are algorithms DP Asynchronous update algorithms These set. state the of sweeps systematic of terms in organized states other of values whatever using whatsoever, order any in states of values the times several updated be may states some of values The available. be to happen however, correctly, converge To once, updated are others of values the before states: the all of values the update to continue must algorithm asynchronous an

DP Asynchronous computation. the in point some after state any ignore can't it update. to states selecting in flexibility great allow algorithms

۱۴-۲ Q-learning

۱۴-۲. Q-LEARNING

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

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

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

فصل ۳

نتایج جدید

در این فصل نتایج جدید به دست آمده در پایان نامه توضیح داده می شود. در صورت نیاز می توان نتایج جدید را در قالب چند فصل ارائه نمود. همچنین در صورت وجود پیاده سازی، بهتر است نتایج پیاده سازی را در فصل مستقلی پس از این فصل قرار داد.

۳-۱ روش های مبتنی بر مدل و بدون مدل

یکی از مهمترین نقاط انشعاب در الگوریتم های 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 پیش بینی می کند.

نکته مثبت اصلی در داشتن مدل این است که به عامل اجازه می دهد با تفکر از قبل، ببیند چه اتفاقی برای طیف وسیعی از گزینه های ممکن رخ می دهد و به صراحت در مورد گزینه های خود تصمیم بگیرد. سپس عامل می تواند نتایج حاصل از برنامه ریزی قبلی را در قالب یک خط مشی بیاموزد یک نمونه مشهور از این روش AlphaZero است. در عمل، اگر دستیابی به مدلی از محیط امکان پذیر و عملی باشد، معمولاً از روش های مبتنی بر مدل استفاده می شود. زیرا می تواند باعث بهبود قابل توجه کارایی نمونه نسبت به روش های بدون مدل شوند.

اصلی ترین نقطه ضعف این روش ها این است که معمولاً یک مدل کامل از محیط در دسترس عامل

نیست و عامل، باید مدل را کاملاً از طریق تجربه یاد بگیرد.

الگوریتم هایی که از یک مدل استفاده می کنند، روش های مبتنی بر مدل و آنهایی که از چنین مدلی استفاده نمی کنند، بدون مدل نامیده می شوند، نامیده می شوند. در حالی که روش های بدون مدل از دستاوردهای بالقوه در بهره وری نمونه با استفاده از مدل چشم پوشی می کنند، اما پیاده سازی و تنظیم آنها آسان تر است. به همین خاطر، روش های بدون مدل از محبوبیت بیشتری برخوردار بوده و به طور گسترده تری نسبت به روش های مبتنی بر مدل توسعه و آزمایش شده اند.

۲-۳ روش های بدون مدل

۳-۳ روش های مبتنی بر ارزش

در روشهای یادگیری تقویتی بدون مدل مبتنی بر ارزش تابع ارزش عمل با استفاده از یک function approximator، مانند شبکه عصبی، نشان داده می شود. فرض کنید $Q(s, a; \theta)$ یک تابع ارزش عمل تقریبی با پارامتر θ باشد الگوریتم های مختلفی برای بروزرسانی θ وجود دارد الگوریتم Q-learning یکی از نمونه های چنین الگوریتمی است که هدف آن تقریب مستقیم تابع عمل-ارزش بهینه $Q^*(s, a) \approx Q(s, a; \theta)$ است

در Q-learning یک مرحله ای، پارامترهای θ از تابع عمل-ارزش با به حداقل رساندن تابع هزینه به شکل مرحله به مرحله آموخته می شوند، به شکلی که تابع هزینه ام- i 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 می شود که s' حالتی است که بعد از حالت s دیده می شود.

۴-۳ روش های مبتنی بر خط مشی

در این قسمت روشهایی را در نظر می گیریم که به جای استفاده از تابع عمل-ارزش یا حالت-ارزش برای دستیابی به خط مشی بهینه، یک خط مشی پارامتریزه^۱ شده را می آموزد که می تواند اقدامات

^۱Parameterized

را بدون استفاده از یک تابع ارزش، انتخاب کند. یک تابع ارزش ممکن است همچنان برای یادگیری پارامترهای خط مشی استفاده شود، اما برای انتخاب اقدام مورد نیاز نیست. ما از نماد $\theta \in \mathbb{R}^d$

برای بردار پارامتر خط مشی استفاده می کنیم. بر خلاف روشهای مبتنی بر ارزش، روشهای مبتنی بر خط مشی مستقیماً تابع خط مشی $\pi_\theta(a|s)$ را تخمین می زنند و پارامترهای θ را با استفاده از صعود گرادیان روی یک مقیاس عملکرد^۲ $J(\pi_\theta)$

یا به طور مستقیم و یا با بیشینه سازی تخمین های محلی از $J(\pi_\theta)$ بروزرسانی می کند. این روش تقریباً همیشه به صورت on-policy عمل می کنند.

همانطور که در ادامه خواهیم دید، می توان از توابع مختلفی برای مقیاس عملکرد J استفاده نمود. یک انتخاب بدیهی $J(\pi_\theta) = \mathbb{E}[R_t]$ است. این روش های تلاش می کنند تابع J را بیشینه کنند

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

$$\nabla J(\theta_t) \text{ که}$$

تخمینی احتمالاتی است که که امیدریاضی آن گرادیان مقیاس عملکرد J را نسبت به پارامترهای خط مشی θ_t تخمین می زند.

به روش هایی که چنین الگویی را برای محاسبه خط مشی بهینه دنبال می کنند، روش های گرادیان خط مشی^۳ می گوئیم. دسته ای از روش های گرادیان خط مشی وجود دارند که تلاش می کنند تخمینی از تابع ارزش را نیز محاسبه کنند. به چنین روش هایی روش های، بازیگر-منتقد (Actor-Critic) گفته می شود که بازیگر (Actor) اشاره به خط مشی آموخته شده و منتقد (Critic) اشاره به تابع ارزش آموخته شده (معمولاً یک تابع حالت-ارزش) دارد.

قضیه ۱-۳ (گرادیان خط مشی)

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

که μ یک توزیع احتمال روی S است که متناسب با تعداد دفعاتی است که حالت s با دنبال کردن

خط مشی π_θ تکرار می شود.

Performance Measure^۲
Policy Gradient^۳

می توان نشان داد [۴]

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

بنابراین در هر گام $\left[R_t \frac{\nabla_\theta \pi_\theta(a|S_t)}{\pi_\theta(a|S_t)} \right]$ یک تخمین گر نااریب از $\nabla J(\pi_\theta)$ خواهد بود. پس می توان 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 روش های بهینه سازی خط مشی به شرح زیر است.

روش های Actor-Critic که الگوریتم ascent Gradient را مستقیماً برای بهینه سازی $J(\pi_\theta)$ به کار می برند.

روش Optimization Policy Proximal که

۳-۵ روش های Actor-Critic

یک نمونه از روش های بهینه سازی خط مشی، خانواده REINFORCE از الگوریتم های یادگیری تقویتی است. [۲]

الگوریتم استاندارد REINFORCE پارامترهای θ را در جهت

$\nabla_\theta \log \pi(a_t|s_t; \theta) R_t$ بروزرسانی می کند که یک تخمین نااریب از $\nabla_\theta \mathbb{E}[R_t]$ است. می توان با کم کردن یک تابع آموخته شده روی حالت ها $b_t(s_t)$ از R_t ، واریانس این تخمین را کاهش داد بطوریکه نااریب باقی بماند. به چنین تابعی پایه گفته می شود. نتیجتاً گرادیان به شکل $\nabla_\theta \log \pi(a_t|s_t; \theta)(R_t - b_t(s_t))$ خواهد بود.

معمولاً از یک تخمین آموخته شده از تابع ارزش به عنوان پایه استفاده می شود $b_t(s_t) \approx V^\pi(s_t)$

که منجر به تخمینی با واریانس بسیار کوچک تر از گرادیان خط مشی می شود در حالیکه تخمین، نااریب باقی می ماند و نتیجتاً عملیات یادگیری با سرعت بیشتری انجام می شود. این روش می تواند به

شکل معماری بازیگر-منتقد^۴ تعبیر شود که خط مشی π بازیگر و پایه b_t منتقد است.

اگر از تخمین یک تابع حالت-ارزش به عنوان پایه استفاده کنیم $b_t(s) = V_{\pi_{\theta_t}}(s)$ عبارت $R_t - b_t(s)$

می تواند به شکل تخمینی از مزیت^۵ عمل a_t در حالت s_t یا $A(a_t, s_t) = Q(a_t, s_t) - V(s_t)$ تعبیر شود. چرا که 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 می شود.

الگوریتم ۱ الگوریتم *Advantage Actor - Critic*

ورودی: یک پارامتریزه سازی مشتق پذیر از خط مشی $\pi_\theta(a|s)$

ورودی: یک پارامتریزه سازی مشتق پذیر از تابع حالت-ارزش $v_\omega(s)$

۱: پارامترهای خط مشی $\theta \in \mathbb{R}^{d'}$ و تابع حالت-ارزش $\omega \in \mathbb{R}^d$ 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: $S' \rightarrow S$

[۱]

Actor-Critic^۴
Advantage^۵

۳-۶ روش TRPO

تعریف ۳-۱ $\rho_\pi : S \rightarrow \mathbb{R}$ 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 [۳]

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

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 [۳] 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.

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

توجه کنید که L_π 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

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

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

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

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

که

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

رابطه بالا نشان می دهد که می توان یک دنباله صعودی از خط مشی ها داشت به طوری که $J(\pi_0) \leq \dots \leq J(\pi_1) \leq J(\pi_2) \leq \dots$ 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 $J(\pi_{i+1}) - J(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i)$ با بیشینه کردن M_i در هر گام می توان اطمینان حاصل کرد که مقیاس عملکرد واقعی J غیرنزولی خواهد بود.

الگوریتم ۲ الگوریتم *PolicyIteration* با مقیاس عملکرد L_π

۱: خط مشی π_0 را مقداردهی اولیه کن

۲: برای $i = 0, 1, \dots$ تکرار کن:

۳: همه ی مزیت های $A_{\pi_i}(s, a)$ را محاسبه کن

۴: $\arg \max_{\pi} [L_{\pi_i}(\pi) - C D_{KL}^m ax(\pi_i, \pi)] \rightarrow \pi_{i+1} =$ که $C = (4\epsilon\gamma)/(1-\gamma)^2$ و $L_{\pi_i}(\pi) =$

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

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.

۷-۳ روش PPO

در روش TRPO دیدیم که بیشینه سازی مقیاس عملکرد L_π ساده تر از مقیاس عملکرد J است ولی در عوض الگوریتم ascent Gradient تنها مجاز به اعمال تغییرات کوچک در خط مشی است. یک راه دیگر برای کنترل تغییرات خط مشی استفاده از تابع clip است. در روش TRPO دیدیم که تابع مقیاس عملکرد، در گام t به شکل زیر است

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

$$r_t(\theta_{new}) = \frac{\pi_{\theta_{new}}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \text{نسبت احتمالات}$$

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 می کنیم

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

که ϵ یک ابرپارامتر^۶ 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 $1 + \epsilon$ و $1 - \epsilon$ تغییر پیدا کرده اند تا تغییرات بزرگ خط مشی 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 می شود تا از تغییرات بزرگ خط مشی جلوگیری شود.

^۶Hyperparameter

۳-۸ روش های Q-learning

خانواده روش های Q-learning تلاش می کنند 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 می کنند. این بهینه سازی تقریباً همیشه به صورت off-policy انجام می شود، به این معنی که هر به روزرسانی می تواند از داده های جمع آوری شده در هر نقطه استفاده کند، بدون در نظر گرفتن نحوه انتخاب عامل برای کشف محیط در هنگام بدست آوردن داده ها. خط مشی مربوطه از طریق ارتباط بین Q^* و π^* بدست می آید: عامل بعد از یادگرفتن تابع $Q_\theta(s, a)$ به طوریکه $Q_\theta(s, a) \approx Q^*(s, a)$ می تواند عمل بهینه در حالت s را به صورت زیر محاسبه کند

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

از جمله الگوریتم های Q-learning می توان به موارد زیر اشاره کرد
روش کلاسیک DQN که حوزه یادگیری تقویتی ژرف را عمیقاً ارتقا بخشید
روش C51 که توزیعی روی عایدی را می آموزد که امیدریاضی آن Q^* است

۳-۹ روش DQN

در جامعه یادگیری تقویتی، به طور معمول یک عملکرد خطی است تقریبی، اما گاهی اوقات از یک تقریب عملکرد غیر خطی به جای آن، مانند یک شبکه عصبی استفاده می شود معمولاً به شبکه عصبی با وزن Q به عنوان یک شبکه Q اشاره می کنیم. شبکه Q را می توان با به کمینه ساختن دنباله ای از توابع هزینه به شکل $L_i \theta_i$ که در گام یک تغییر می کند، آموزش داد

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

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

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

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

often is it gradient. above the in expectations full the computing than Rather
gradient stochastic by function loss the optimise to expedient computationally
expectations the and time-step, every after updated are weights the If descent.
emulator the and π distribution behaviour the from samples single by replaced are
Note Q -learning familiar the at arrive we then respectively, E
directly task learning reinforcement the solves it model-free: is algorithm this that
estimate an constructing explicitly without E, emulator the from samples using
a: $Q(s, \max_a = a$ strategy greedy the about learns it off-policy: also is It E. of
exploration adequate ensures that distribution behaviour a following while ϵ),
an by selected often is distribution behaviour the practice, In space, state the of
selects and $1 - \epsilon$ probability with strategy greedy the follows that strategy -greedy
 ϵ . behavior policy Q - learning. probability with action random a

توجه داشته باشید که این یک الگوریتم بدون مدل است: این کار وظیفه یادگیری تقویتی را مستقیماً
با استفاده از نمونه های شبیه ساز E بدون ساختن صریح تخمین E حل می کند. در مورد استراتژی
حریص $Q(s, \max_a = a$ ، یاد می گیرد ، در حالی که توزیع رفتاری را دنبال می کند که کاوش
کافی در فضای دولت را تضمین می کند. در عمل ، توزیع رفتار اغلب توسط یک استراتژی Greedy
انتخاب می شود که استراتژی حریصانه را با احتمال $1 - \epsilon$ دنبال می کند - و یک اقدام تصادفی با احتمال

ϵ

stochastic gradient descend^y

الگوریتم ۳ الگوریتم Q-learning با replay Experience

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

۳-۱۰ روش C۵۱

۳-۱۱ مقایسه روش بهینه سازی خط مشی و Q-learning

نقطه قوت اصلی روش های بهینه سازی خط مشی، اصولی بودن آنهاست، به این معنا که شما مستقیماً چیزی که می خواهید را بهینه سازی می کنید. در نتیجه این روش ها قابل اتکا و باثبات هستند. در مقابل، روشهای Q-learning با یادگیری تابع، Q مقیاس عملکرد را به طور غیر مستقیم بهینه می کند. حالت های زیادی برای این نوع یادگیری وجود دارد که به شکست منتهی می شود، بنابراین این روش ها ثبات کمتری دارند. [۱] اما، روش های Q-learning این مزیت را دارند که در هنگام کار، به طور قابل ملاحظه ای کارآمد هستند، زیرا آنها می توانند از داده ها به طور موثرتری نسبت به تکنیک های بهینه سازی خط مشی استفاده کنند.

تعامل بین بهینه سازی خط مشی و Q-learning. بهینه سازی خط مشی و Q-learning ناسازگار نیستند (و به نظر می رسد تحت برخی شرایط، معادل آن باشد)، و طیف وسیعی از الگوریتم ها وجود دارد که بین دو حد این طیف زندگی می کنند. الگوریتم هایی که در این طیف زندگی می کنند قادرند با دقت بین نقاط قوت و ضعف طرفین معامله کنند. مثالها شامل

DDPG، الگوریتمی است که همزمان با استفاده از هر یک برای بهبود دیگری، یک خط مشی قطعی و یک تابع Q را یاد می گیرد، و SAC، نوعی که از خط مشی های تصادفی، تنظیم آنتروپی^۸ و چند ترفند دیگر برای تثبیت یادگیری و کسب امتیاز بالاتر از DDPG در معیارهای استاندارد استفاده می کند.

۳-۱۲ روش DDPG

Here we combine the actor-critic approach with insights from the recent success of Deep Q Network (DQN) (Mnih et al., 2013; 2015). It was the first time that a non-linear function approximator was able to learn value functions that generally believed using large non-linear functions was unstable and difficult.

^۸entropy regularization

inno- two to due way robust and stable a in approximators function such using
buffer replay a from samples with off-policy trained is network the .۱ vations:
tar- a with trained is network the .۲ samples: between correlations minimize to
In backups. difference temporal during targets consistent give to network Q get
(Ioffe normalization batch with along ideas, same the of use make we work this
learning. deep in advance recent a, (۲۰۱۵ Szegedy,

الگوریتم ۴]

الگوریتم [DDPG 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. پارامترهای توابع هدف μ' و Q' را با وزن های حافظه تکرارها R را بساز برای هر اپیزود $\theta_\mu \longleftrightarrow \theta_{\mu'}$ و $\theta_Q \longleftarrow \theta_{Q'}$ مقداردهی اولیه کن
 $1 \dots M$ یک تابع نویز تصادفی \mathbb{N} بساز حالت اولیه S_1 را مشاهده کن برای $t = 1 \dots T$
عمل $a_t = \mu(s_t; \theta_\mu) + \mathbb{N}_t$ را بر اساس خط مشی فعلی و نویز اکتشاف، انتخاب کن و حالت بعدی S_{t+1} و پاداش R_t را مشاهده کن. تجربه (s_t, a_t, r_t, s_{t+1}) را در انبار تجربه R ذخیره کن
یک نمونه به اندازه N از تجربه های (s_i, a_i, r_i, s_{i+1}) از انبار تجربه R انتخاب کن
وزن های منتقد θ_Q را با در نظر گرفتن تابع هزینه $L = \frac{1}{N} \sum_i (y_i - Q(S_i, A_i; \theta_Q))^2$ به روزرسانی کن
وزن های بازیگر θ را با استفاده از گرادیان خط مشی 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 روزرسانی کن وزن های توابع هدف را به شکل

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

به روزرسانی کن

۱۳-۳ روش SAC

۱۴-۳ روش های مبتنی بر مدل

of clusters easy-to-define of number small a aren't there RL, model-free Unlike
models, using of ways orthogonal many are there RL: model-based for methods
the case, each In exhaustive, from far is list the but examples, few a give We'll
learned, or given be either may model

repre- explicitly never approach basic most The Planning, Pure Background:
model-predictive like techniques planning pure uses instead, and policy, the sends
envi- the observes agent the time each MPC, In actions, select to (MPC) control
where model, the to respect with optimal is which plan a computes it ronment,
the after time of window fixed some over take to actions all describes plan the
planning the by considered be may horizon the beyond rewards (Future present,
executes then agent The function.) value learned a of use the through algorithm
a computes It it, of rest the discards immediately and plan, the of action first the
using avoid to environment, the with interact to prepares it time each plan new
horizon, planning shorter-than-desired a with plan a from action an

some on models environment learned with MPC explores work MBMF The
follow- straightforward A Iteration, Expert RL, deep for tasks benchmark standard
the of representation explicit an learning and using involves planning pure to on

$\pi_{\theta}(a|s)$. The agent uses a planning algorithm (like Monte Carlo Tree Search) in the model, generating candidate actions.

play to networks neural deep train to approach this uses algorithm ExIt The
for Augmentation Data approach, this of example another is AlphaZero Hex.
Q- or policy a train to algorithm RL model-free a Use Methods, Model-Free
updating in ones fictitious with experiences real augment (۱ either but function,
agent, the updating for experience fictitious only use (۲ or agent, the

fictitious with experiences real augmenting of example an for MBVE See
 experience fictitious purely using of example an for Models World See ones.
 Planning Embedding dream.” the in “training call they which agent, the train to
 directly procedure planning the embeds approach Another Policies. into Loops
 information side become plans complete that subroutine—so a as policy a into
 model– standard any with policy the of output the training policy—while the for
 to learn can policy the framework, this in that is concept key The algorithm. free
 problem, a of less bias model makes This plans. the use to when and how choose
 learn simply can policy the states, some in planning for bad is model the if because
 it. ignore to

imagina– of style this with endowed being agents of example an for I۲A See

tion.

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 ارضا پذیری
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 توزیع شده	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