به نام خدا

هوش مصنوعی

نیم‌سال دوم ۹۷ـ۹۸

دانشکده مهندسی کامپیوتر

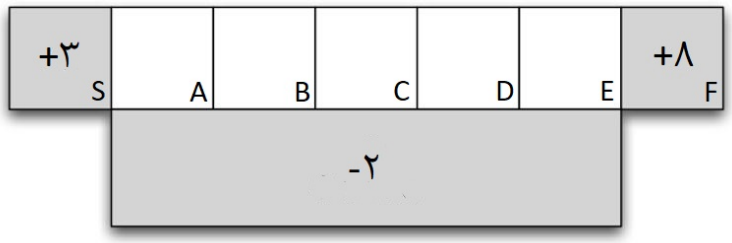
استاد: دکتر سلیمانی

تمرین چهارم (۳۰+۱۰۵ نمره)

مهلت ارسال تمرین: ۲ اردیبهشت ماه

تمرین‌های نظری (۴۰ نمره)

سوال اول(۱۲ نمره)



شکل ۱: محیط MDP مسئله

محیط MDP بالا را در نظر بگیرید. در این محیط، تنها دو حرکت چپ و راست داریم. شروع حرکت ایجنت از خانه A است. جایزه رسیدن به هر استیت بر روی آن خانه نوشته شده‌است. بقیه خانه‌ها و همچنین هر اکشن، جایزه‌ای ندارند. در هر حرکت ایجنت به سمت چپ یا راست، به احتمال p همان حرکت انجام شده و به احتمال $1-p$ ایجنت به روی خانه‌های پایین که گودال است، می‌افتد. خانه‌های خاکستری، خانه پایانی می‌باشند و رفتن به آنها به معنای پایان episode است.

۱. فرض کنید $\gamma = 1$ است. در این صورت، به ازای چه مقادیری از p حرکت بهینه در خانه A رفتن به راست خواهد بود؟ (۲نمره)

۲. حال فرض کنید $p = 1$. در این صورت، به ازای چه مقادیری از γ حرکت بهینه در A رفتن به راست خواهد بود؟ (۲نمره)

۳. با استفاده از الگوریتم value iteration مقادیر حالت‌ها را پس از ۱، ۲ و ۳ بار بروزرسانی، بر حسب p و γ بدست بیاورید. مقادیر اولیه را صفر در نظر بگیرید. (۴نمره)

۴. در مقادیر بدست آمده در بالا، $p = 0.8$ و $\gamma = 0.9$ را جایگزین کرده و برای مقادیر بدست آمده در آخرین شمارش، با استفاده از one step expectimax سیاست بهینه را بدست آورید. (۴نمره)

سوال دوم(۱۱ نمره)

گاهی اوقات،در مسائل MDP تابع reward به‌صورت $R(s,a,s')$ یا $R(s,a)$ یا $R(s)$ تعریف می‌شود که s و s' و a به ترتیب state مبدا، state مقصد و action انجام شده هستند.

۱. برای هریک از این فرمول بندی‌ها، روابط bellman را بنویسید.(۳نمره)

۲. نشان دهید که چگونه یک مسئله MDP با تابع جایزه $R(s,a,s')$ می‌تواند به صورت یک مسئله MDP با تابع جایزه $R(s,a)$ تبدیل شود بطوری که سیاست بهینه‌ی مسئله اول دقیقا متناظر با سیاست بهینه‌ی مسئله‌ی دوم باشد.(۴نمره)

۳. حال همین تبدیل را برای MDP با $R(s,a)$ به MDP با $R(s)$ انجام دهید. (۴نمره)

سوال سوم (۱۷نمره)

۱. ثابت کنید در Markov Decision Process محاسبه‌ی utility به صورت discounted [۱] ، stationary preference [۲] است اما اگر utility را برابر با بیشینه reward قابل کسب در دنباله در نظر بگیریم دیگر این خاصیت برقرار نیست. (۳نمره)

۲. به طور دقیق ثابت کنید که در الگوریتم Value iteration بعد از اجراهای متوالی، مقادیر V_i^* ها همگرا می شوند. (۴نمره)

۳. آیا در Q-learning مبتنی بر ویژگی‌ها که مقدار Q ها با استفاده از چندین feature بیان می شود، همانند Q-learning عادی بعد از تعداد گام کافی به استراتژی اپتیمال همگرا می شود یا نه؟ دلیل را توضیح دهید. (۲نمره)

۴. فرض کنید یک بازی Zero Sum نوبتی دو نفره را به صورت MDP فرمول بندی کرده ایم. $U_A(s)$ و $U_B(s)$ را به ترتیب مقدار utility وضعیت s برای فرد A و B (که مقدارشان قرینه ی هم است)درنظر بگیرید. فرض کنید همه ی reward ها و utility ها از دید فرد اول محاسبه می شود. روابط Bellman را برای این مسئله بنویسید و جزئیات اجرای الگوریتم value-iteration و شرط خاتمه ی آن را بیان کنید. توضیح دهید که چرا حل این مسئله به این روش، نسبت به روش minmax می تواند بهتر باشد. (۸ نمره)

---

[۱] $U([r_0, r_1, r_2, ...]) = r_0 + \gamma r_1 + \gamma^2 r_2 + ...$

[۲] $[a_1, a_2, ...] \succ [b_1, b_2, ...] \iff [r, a_1, a_2, ...] \succ [r, b_1, b_2, ...]$ یعنی اگر دنباله پاداش A به دنباله پاداش B ترجیح داده شود و عضو اول هردو دنباله یک مقدار باشد، آنگاه دنباله پاداش A' به دنباله پاداش B' ترجیح داده شود و برعکس . (A' همان دنباله پاداش A با حذف عضو اول است و مشابه آن B')

تمرین‌های عملی (۳۰+۶۵ نمره)

سوال اول (۴۰ نمره)

در این سوال، شما باید با استفاده از الگوریتم‌هایی که یاد گرفته‌اید، به حل کردن محیط Taxi-v۲ از مجموعه محیط‌های OpenAI Gym بپردازید. توضیحات مربوط به این محیط را می‌توانید در لینک‌های داده‌شده ببینید. در اینجا بصورت مختصر، محیط را شرح می‌دهیم. این بازی از یک نقشه ۵ در ۵ تشکیل شده‌است که ۴ لوکیشن به نام‌های R ، G ، B و Y دارد. در یکی از این ۴ تا، مسافر وجود دارد و یکی از ۳ تای باقی مانده، مقصد مسافر است. علامت | نشان دهنده دیوار است. بدین معنی که تاکسی نمی‌تواند از آن رد شود. علامت : به معنی یک مسیر باز است که تاکسی می‌تواند از آن عبور کند. هدف محیط این است که تاکسی مسافر خود را از مکانی که در آن است، سوار کرده و در مقصد پیاده کند. در این بازی ۶ اکشن داریم. بالا، پایین، چپ، راست، سوار کردن و پیاده کردن. به ازای هر حرکت، تاکسی امتیاز ۱ـ را دریافت کرده و به ازای هر پیاده کردن یا سوار کردن بی مورد، امتیاز ۱۰ـ را دریافت می‌کند. همچنین اگر مسافر را در مکان مورد نظر پیاده کند، امتیاز ۲۰ را دریافت می‌کند. در این بازی ۵۰۰ استیت داریم که دارای اطلاعات مکان تاکسی، مکان مسافر، بودن یا نبودن مسافر در تاکسی و مقصد مسافر می‌باشد. شکل محیط در پایین آمده‌است. این شکل در تمام episode های بازی به همین صورت است اما مکان اولیه تاکسی می‌تواند هرجایی از صفحه بازی باشد. در صورت وجود سوال، می‌توانید به کد Taxi.py که در اختیارتان قرار گرفته‌است، مراجعه کنید و از توابع موجود بهره ببرید. توجه داشته باشید که ایجنت رندوم در این محیط، امتیاز منفی زیادی دریافت می‌کند و حتی ممکن است نتواند بازی را تمام کند و تا مدت زمان طولانی‌ای، بازی تمام نشود و بهترین امتیاز ممکن نیز ۲۰ است که تقریبا رسیدن به آن غیرممکن است. چرا که هر حرکت ۱ امتیاز جریمه دارد.

شکل ۲: محیط برنامه. خانه زرد رنگ مکان تاکسی را نمایش می‌دهد که می‌تواند تغییر کند.

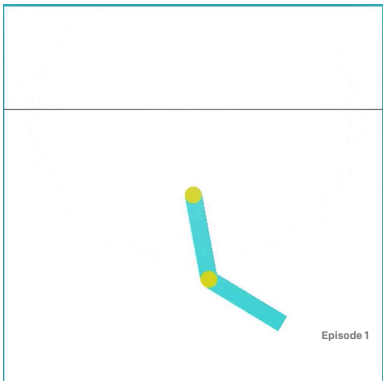
- ابتدا با استفاده از value iteration این مسئله را حل کنید. محیط داده شده را known در نظر بگیرید. دادن ورودی به محیط و چگونگی کار با آن توسط یک ایجنت رندوم و همچنین دریافت اطلاعات محیط ($T(s,a,s')$ و $R(s,a,s')$) نیز در کد Taxi.py در اختیارتان قرار گرفته‌است. سیاست بدست آمده را با در یک آرایه به طول ۵۰۰ که هر خانه آن نشان دهنده استراتژی در آن

استیت است، قرار داده و با استفاده از کتابخانه pickle با نام value_iteration_policy.pkl ذخیره کنید.(۱۵ نمره)

- بار دیگر این مسئله را با policy iteration حل کنید. در هر دو حالت ابتدا الگوریتم را با $\gamma = 0.99$ و بار دیگر با $\gamma = 1$ اجرا کنید و نتیجه و علت آن را گزارش کنید. در نهایت، جواب در این دو حالت را با بصورت تقریبی مقایسه کنید. آیا جواب‌ها در این دو حالت تفاوت زیادی می‌کنند؟ همچنین از لحاظ زمانی نیز این دو الگوریتم را مقایسه کنید. همچنین سیاست بدست آمده را با در یک آرایه به طول ۵۰۰ که هر خانه آن نشان دهنده کنش بهینه در آن وضعیت است (یک عدد بین ۰ تا ۵)، قرار داده و با استفاده از کتابخانه pickle با نام policy_iteration_policy.pkl ذخیره کنید.(۲۵ نمره)

سوال دوم (۳۰+۲۵ نمره)

در این سوال شما باید الگوریتم Q-learning را روی محیط Acrobot-v۱ از مجموعه محیط‌های OpenAI Gym پیاده سازی کنید. در این محیط دو تکه چوب همسان مطابق شکل سه توسط یک مفصل به هم وصل شده‌اند و چوب اول توسط مفصلی ثانویه به جایی ثابت وصل است. در هر لحظه agent می تواند به مفصل مابین دو چوب گشتاور -1 یا 0 یا $+1$ وارد کند.همچنین agent در هر لحظه سینوس و کسینوس هردو زاویه ی $\theta 1$ و $\theta 2$ و همچنین سرعت زاویه ای آنها مطلع است.(در مجموع ۶ داده) $\theta 1$ و $\theta 2$ به ترتیب زاویه‌ی چوب بالایی با سطح ثابت و زاویه دوچوب است. به بیان دقیقتر $\theta 1$ وقتی صفراست که چوب بالایی مستقیم به سمت پایین اشاره کند. $\theta 2$ هم نسبت به چوب اولی در نظر گرفته می شود، یعنی وقتی صفر است که هر دوچوب هم جهت باشند. این محیط با در نظر گرفتن وزن برای هر دو چوب و مفصل ها، لحظه به لحظه با توجه به قوانین فیزیک مکانیک، وضعیت کل را شبیه سازی می کند. در این محیط هدف agent رساندن سر پایینی چوب زیرین، به حداقل ارتفاعی معین است؛باید حداقل به اندازه طول یک چوب، بالاتر از مفصل متصل به سطح ثابت بیاید. برای اینکار ۵۰۰ گام فرصت دارد، هرگام که این هدف محقق نشود،مقدار پاداش -1 کسب می شود و زمانی که سر چوب به ارتفاع بالا رسید، اپیزود تمام می شود.پس پالیسی بهتر، سعی می کند که هرچه زودتر اینکار را انجام دهد. کد نیمه آماده‌ی acro.py برای آشنایی شما با این محیط در اختیارتان قرار گرفته است.



شکل ۳: نمایی از محیط Acrobot که دو چوب توسط مفصلی به هم وصلند و چوب بالایی توسط مفصلی ثانویه به محلی ثابت متصل شده.

- با استفاده از الگوریتم Q-learing و با رفتار کاملا unkown نسبت به محیط پالیسی بهینه را یادگیری کنید.مقدار هایپر پارامترها رابرابر با آنچه که در کد نیمه آماده، آمده است قراردهید. پالیسی بهینه را با استفاده از کد و مطابق اسم پیشفرضی که ذکرشده (q-saved.npy) ذخیره کنید. پالیسیتان برای نمره دهی ۱۰۰۰ اپیزود اجرا می شود و باید میانگین امتیازش بیشتر از 270− باشد تا امتیاز کامل را کسب کند.(۲۵ نمره)
- حال می توانید در تابع env-state-to-Q-state تغییر ایجاد کنید و تبدیل حالات پیوسته به گسسته را طوری دیگر انجام دهید، تعداد state هارا بیشتر کنید یا هایپرپارامتر هارا تغییر دهید، هرچه پالیسیتان میانگین امتیاز بیشتری کسب کند، نمره‌ی امتیازی بیشتری کسب می کنید.(حداکثر ۶ نمره امتیازی)
- در این قسمت به تخمین تابع Q به کمک تابع پارامتریک بپردازید و سعی کنید با استفاده از Q-learning مبتنی بر feature ، میانگین امتیاز agent را به حداقل 270− برسانید. توصیه می شود برای این قسمت، به source-code مختصر این environment و بخصوص کامنت اولیه ی آن نگاهی بندازید تا آگاهی بیشتری نسبت به داده های هر state داشته باشید. (۲۴ نمره امتیازی)