دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده مهندسی برق و کامپیوتر

بهبود یادگیری عامل یادگیرنده اجتماعی به کمک مشاهده رفتار عامل‌های دیگر

پایان‌نامه برای دریافت درجه کارشناسی

در رشته مهندسی برق گرایش کنترل

نام

نیما زمان پور

شماره دانشجویی

۸۱۰۱۹۸۴۰۷

استاد راهنما:

مجید نیلی احمد آبادی

تیرماه ۱۴۰۳


بسم الله الرحمن الرحيم

تعهدنامه اصالت اثر

باسمه تعالی

اینجانب نیما زمان پور تائید می کنم که مطالب مندرج در این پایان نامه حاصل تلاش اینجانب است و به دستاوردهای پژوهشی دیگران که در این نوشته از آنها استفاده شده است مطابق مقررات ارجاع گردیده است. این پایان نامه قبلاً برای احراز هیچ مدرک هم سطح یا بالاتر ارائه نشده است.

نام و نام خانوادگی دانشجو :

نیما زمان پور

امضای دانشجو :

نیما زمان پور – ۲۲/۰۳/۱۴۰۳

تشکر و قدردانی[1]:

.....

با سپاس فراوان از جناب دکتر نیلی که در مراحل طراحی و به ثمر رساندن پروژه همواره راهنمای توانمندی برای اینجناب بودند. همچنین از جناب آقای علیرضا رشیدی لاله که بدون پشتیبانی ایشان به سرانجام رساندن این تحقیق بسیار مشکل تر می‌شد. در آخر نیز از لپتاپم که شبانه روز بی‌وقفه مشغول پردازش تحقیق من بود قدردانی می‌کنم.

[1] Acknowledgements-

چکیده[1]

آموزش عامل یادگیری تقویتی همواره یک فرآیند دشوار و زمان بر بوده‌است. بسیاری از محیط ها دارای ابعاد بالا بوده و یا سیستم های واقعی هستند که به دلیل نداشتن قابلیت مدل سازی تعامل با آن ها بسیار کند صورت می‌پذیرد. در این سیستم ها وجود عوامل دیگر، یک فرصت عالی برای استخراج تجربه و دانش آن‌ها در اختیار ما می‌گذارد. تا سرعت و کیفیت یادگیری افزایش یابد. اما دیگر عوامل ممکن است در سطح تخصص و توابع هدف با هم فرق داشته باشند. از این مهم تر تمایلی به اشتراک گذاری سیاست و پاداش خود نداشته باشند.

برای حل این مسئله ما یک روش جدید ابداع کرده‌ایم که به تدریج از تمامی عامل های موجود در محیط یک مدل می‌سازد. سپس، به هر عامل یک امتیاز، بر اساس عملکرد مدل آن عامل تخصیص می‌دهد. و به عوامل با امتیاز بیشتر، شانس بیشتر برای تعامل با محیط می‌دهد که برای عامل ما expert trajectory با کیفیتی برای افزایش سرعت یادگیری فراهم می‌سازد.

نتایج آزمایشات این روش نشان می‌دهد که علاوه بر اینکه این روش سرعت یادگیری عامل را تا چند برابر سریع می‌کند. در حین این یادگیری نیز به دلیل استفاده از سیاست مدل عامل متخصص به جای سیاست خود، همواره پاداش بالایی را حتی در قسمت های اولیه آموزش کسب می‌کند.

واژه‌های کلیدی: یادگیری تقویتی، عامل اجتماعی، تقلید رفتار، الگوریتم بازیگر-نقاد نرم، یادگیری Q محافظه‌کارانه

[1] Abstract

فهرست مطالب

فصل ۱: مقدمه و بیان مساله ۱

۱-۱- مقدمه ۱

۱-۲- شرح مسئله تحقیق ۱

۱-۳- اهداف و آرمان‌های کلی تحقیق ۲

۱-۴- روش انجام تحقیق ۲

۱-۵- ساختار پایان‌نامه ۳

فصل ۲: مفاهیم اولیه و پیش زمینه ۴

۲-۱- الگوریتم Policy Gradient ۴

مقدمه ۴

هدف اصلی الگوریتم ۵

روش بروز رسانی ۵

2-2- الگوریتم Soft Actor-Critic ۶

مقدمه ۶

هدف بهینهسازی ۶

تعریف شبکههای الگوریتم ۶

روش آپدیت الگوریتم ۷

2-3- الگوریتم Conservative Q-Learning ۹

مقدمه ۹

عبارت محافظه کارانه ۹

۲-۴- روشBehavior Cloning ۱۰

فصل ۳: ساختار روش پیشنهادی ۱۰
-3-1 مقدمه ۱۰
-3-2 مدلسازی عاملهای اجتماعی ۱۱
-3-3 کارکرد مدلها در روش ۱۱
۴-۳- معیار امتیازدهی ۱۱
-3-5 الگوریتم یادگیری عامل ۱۲
-3-6 مزایای مدل ۱۲
فصل ۴: پیادهسازی و ارزیابی نتایج ۱۴
۱-۴- مقدمه ۱۴
۲-۴- محیط پیادهسازی ۱۴
۳-۴- عامل شخصی ۱۵
۴-۴- عاملهای اجتماعی ۱۶
۵-۴- نتایج ۱۷
آزمایش ۱: ۱۷
آزمایش ۲ ۱۸
آزمایش ۳ ۲۱
آزمایش ۴ ۲۴
آزمایش ۵ ۲۶
۶-۴- خلاصه و جمعبندی ۲۹
فصل ۵: جمعبندی، نتیجهگیری و پیشنهادها ۳۰
۱-۵- نتیجهگیری ۳۰

۱-۱-۵- محدودیتها ۳۱

۲-۱-۵- پیشنهادها ۳۱

فصل ۶: مراجع ۳۲

پیوست‌ها ۳۴

فهرست شکل‌ها

شکل ۱ شبهکد روش Policy Gradient ۶
شکل ۲ شبکه کد الگوریتم SAC (ورژن اولیه ۸
شکل ۳ شهود روش CQL در کم کردن مقدار Q-value ها ۹
شکل ۴ نمای محیط شبیهسازی ۱۵
شکل ۵ نمودار پاداش عامل در طول آموزش ۱۷
شکل ۶ نمودار پاداش عامل در حالت ارزیابی ۱۸
شکل ۷ نمودار پاداش دسته در طول آموزش ۱۹
شکل ۸ نمودار پاداش عامل در حالت ارزیابی ۱۹
شکل ۹ نمودار امتیاز مدلها در طول آموزش ۲۰
شکل ۱۰ نمودار احتمال انتخاب مدلها در طول آموزش ۲۰
شکل ۱۱ نمودار پاداش دسته در طول یادگیری ۲۵
شکل ۱۲ نمودار احتمال انتخاب مدلها در طول یادگیری ۲۵
شکل ۱۳ نمودار پاداش عامل در حالت ارزیابی ۲۶
شکل ۱۴ نمودار امتیاز مدلها در طول یادگیری ۲۶
شکل ۱۵ نمودار پاداش دسته در طول آموزش ۲۷
شکل ۱۶ نمودار پاداش عامل در حالت ارزیابی ۲۸
شکل ۱۷ نمودار احتمال انتخاب مدلها در طول ارزیابی ۲۸
شکل ۱۸نمودار امتیاز مدلها در طول آموزش ۲۹

فهرست جدول‌ها

جدول 1 جدول جزئیات شبیهسازیها..... ۱۶
جدول ۲ جدول امتیاز مدل خبره دشوار بر حسب تعداد قسمتهای در دسترس..... ۲۱

فهرست علائم اختصاری

lr	Learning Rete
TAU	Q Network Soft Update Coeff
CQL	Conservative Q-Learning
SAC	Soft Actor-Critic
SA	Social Agent
BC	Behavior Cloning
π_θ	Agent Policy Parameterized with θ

فصل ۱

فصل ۱:

مقدمه و بیان مساله

۱-۱- مقدمه

یادگیری تقویتی در زمینه آموزش عامل‌های هوشمند برای تصمیم‌گیری مستقل در محیط های پیچیده، نوید زیادی نشان داده‌است. با این حال، مهمترین چالش این زمینه، زمان بسیار زیاد مورد نیاز برای تعامل با محیط برای کسب سیگنال های پاداش و یادگیری سیاست بهینه می‌باشد. در محیط هایی که هزینه زمانی یا مالی کسب تجربه فردی، کمتر از هزینه پردازش سخت افزاری است؛ وجود دیگر عامل‌های اجتماعی، این فرصت را می‌دهد که از استخراج تجربه و دانش آن‌ها برای افزایش سرعت یادگیری استفاده کرد.

۱-۲- شرح مسئله تحقیق

در این پروژه قصد داریم به کمک مشاهده رفتاره عامل های اجتماعی دیگر در محیط، بدون داشتن هر گونه ارتباط با عامل که به قصد انتقال پاداش، وزن های شبکه، و یا سیاست عامل منجر شود؛ تجربه عامل اجتماعی را استخراج نموده و با ایجاد یک مدل از آن عامل، و شناسایی سطح تخصص آن، به

بهره‌برداری از دانش عامل برای تعامل با محیط به جای عامل کم‌تجربه ما و نیز تشکیل یک داده‌گان از تجربیان عامل متخصص بپردازیم. این روش گام بلندی در راستای افزایش سرعت یادگیری عامل و کاهش تعاملات کم‌بهره برمی‌دارد.

۳-۱- اهداف و آرمان‌های کلی تحقیق

مهمترین هدف این پروژه، افزایش سرعت یادگیری و کاهش تعامل با محیط با بهره‌برداری از مشاهده رفتار عامل‌های اجتماعی است. بطور دقیق‌تر، انگیزش ما دستیابی به اهداف زیر است:

- توانایی ایجاد مدل از عامل اجتماعی: روش ما باید بتواند با داشتن دوتایی های (حالت، عمل) به‌شیوه آنلاین و آفلاین، توانایی ساخت مدلی از سیاست عامل اجتماعی داشته باشد.
- توانایی در شناسایی سطح دانش عامل اجتماعی: روش باید بتواند به خوبی به‌شیوه کمی سطح دانش عامل اجتماعی را متناسب با تابع هدف خود ارزیابی نموده، و آن را بشکل پویا آپدیت کند.
- انتخاب بهینه بین عامل‌ها: روش باید بتواند صرف نظر از روند یادگیری عامل خودی، در هر زمان، بهترین عامل اجتماعی را برای تعامل با محیط بر اساس یک معیار انتخاب کند.
- قابلیت یادگیری آفلاین (offline RL): عامل باید بتواند علاوه بر یادگیری به کمک سیاست خود، یادگیری از طریق تجربیات عاملی که با یک سیاست متفاوت عمل می‌کند نیز داشته باشد.

۴-۱- روش انجام تحقیق

در این تحقیق ابتدا الگوریتم های مورد نیاز برای مدل سازی، یادگیری عامل و یادگیری آفلاین پیاده سازی می‌شوند. سپس از عامل‌های از پیش آموزش داده شده در سایت Hugging Face در سطوح مختلف در نقش عامل‌های اجتماعی استفاده می‌کنیم. سپس از محیط های کتابخانه Gymnasium برای ارزیابی روش پیشنهادی استفاده می‌کنیم.

۵-۱- ساختار پایان‌نامه

در فصل دوم، شامل بررسی تعاریف اساسی مربوط به حوزه‌ی یادگیری تقویتی ، مفاهیم اولیه و اجزای اساسی الگوریتم های SAC, CQL می‌پردازیم.

فصل سوم در برگیرنده‌ی توضیح مربوط به مدل پیشنهادی و پیاده سازی شده است.

در فصل چهارم آزمایش‌های انجام شده بر روی مدل را شرح داده و نتایج آن را تفسیر می‌کنیم.

در نهایت، در فصل پنجم، نتیجه‌گیری‌های حاصل شده در این تحقیق ارائه خواهد شد. و محدودیت‌ها مورد بحث قرار می‌گیرد و پیشنهادهایی برای ادامه‌ی مسیر به علاقمندان این حوزه‌ی ارائه خواهد شد.

فصل ۲

فصل ۲: مفاهیم اولیه و پیش زمینه

در فصل پیش رو مقدمات، مفاهیم اولیه و پیش‌زمینه‌هایی را که جهت درک هر چه بهتر موضوع‌های مطرح شده در این پایان‌نامه مورد نیاز است، ارائه می‌شود. در این فصل مقدماتی در مورد الگوریتم‌های Policy Gradient, Soft Actor Critic, Conservative Q-Learning ارائه می‌شود

۲-۱- الگوریتم Policy Gradient

مقدمه

الگوریتم Policy Gradient یکی از روش‌های اصلی در یادگیری تقویتی است. این روش به جای تخمین ارزش حالت‌ها ($V(s)$) یا حالت-عمل‌ها ($Q(s,a)$) بصورت مستقیم یک سیاست (policy) $\pi_\theta(a|s)$ را به قصد پیدا کردن عمل بهینه به ازای هر حالت آموزش می‌دهد. این روش با پارامتریزه کردن یک تابع به عنوان سیاست عامل برای حالات عمل-پیوسته نیز جوابگو است. که یک مزیت نسبت به روش Q-Learning محسوب می-شود.

در یک محیط، معمولا مجموع پاداش عامل بصورت تخفیف دار در طول زمان بر اساس یک افق[1]

[1] Horizon

محاسبه می‌شود. اگر r_i پاداش عامل در هر حالت و γ ضریب تخفیف باشد مجموع پاداش R_t بصورت زیر محاسبه می‌شود:

$$R_t = \sum_{i=0}^{t} r_i * \gamma^i$$

هدف اصلی الگوریتم

در این الگوریتم هدف پیدا کردن پارامتر های θ است. که شکلی که میانگین مجموع پاداش عامل در طول زمان بهینه شود.

$$\max_{\theta} J(\theta) = \mathbb{E}_{\theta^{\tau\sim\pi}}[R(\tau)]$$

روش بروز رسانی

برای بهینه سازی تابع هزینه $J(\theta)$ ، از آن گرادیان نسبت به پارامتر های θ گرفته و در جهت بیشترین شیب[1] پارامترها را آپدیت می‌کنیم.

$$\mathbb{E}_{\theta^{\tau\sim\pi}}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R_t\right] = \nabla_\theta J(\theta)$$

$$\alpha\nabla_\theta J(\theta) + \theta \leftarrow \theta$$

[1] Steepest ascent

Monte-Carlo Policy Gradient (REINFORCE)

- Update parameters by stochastic gradient ascent
- Using policy gradient theorem
- Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

function REINFORCE
  Initialise θ arbitrarily
  for each episode $\{s_1, a_1, r_2, ..., s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**
    for $t = 1$ to $T - 1$ **do**
      $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$
    end for
  end for
  return θ
end function

شکل ۱ شبه‌کد روش Policy Gradient

۲-۲- الگوریتم Soft Actor-Critic

مقدمه

الگوریتم SAC (Soft Actor-Critic) یکی از پیشرفته ترین الگوریتم های یادگیری تقویتی است. که توانایی فوق‌العاده‌ای در بهبود عملکرد و پایداری در محیط‌های پیوسته دارد. مهمترین ویژگی این الگوریتم، داشتن عبارت Entropy Regularization است. که به افزایش تنوع در سیاست یادگرفته شده کمک می‌کند تا عامل در بهینه‌های محلی به دام نیوفتد.

هدف بهینه‌سازی

هدف این الگوریتم پیدا کردن سیاست π^* است. بگونه‌ای که میانگین زیر ماکسیمم شود:

$$\pi^* = arg \max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t, a_t, s_{t+1}) + \alpha H\big(\pi(\cdot\,|s_t)\big) \right) \right]$$

که در آن تابع H آنتروپی و آلفا ضریب اهمیت آن می‌باشد.

تعریف شبکه‌های الگوریتم

در این الگوریتم یک شبکه سیاست و چهار شبکه برای تخمین ارزش Q وجود دارد. که دو شبکه برای آپدیت نرم مقدارهای Q استفاده می‌شود. و مینیمم دو شبکه دیگر نیز برای کاهش اثر overestimation در شبکه‌های Q استفاده می‌شود. در نسخه اولیه این الگوریتم، یک شبکه برای تخمین ارزش V نیز وجود داشت که به دلیل ساده‌تر کردن الگوریتم و کاهش خطا بعدا حذف شده. و در این تحقیق نیز استفاده نشده. همچنین در نسخه‌های جدیدتر ضریب آنتروپی پویا شده که در این تحقیق نیز همین‌گونه است.

شبکه Q:

$$Q^{\pi}(s,a) = \mathbb{E}_{s'\sim P, a'\sim\pi}\left[R(s,a,s') + \gamma\left(Q^{\pi}(s',a') + \alpha H\big(\pi(\cdot\,|s')\big)\right)\right]$$
$$= \mathbb{E}_{s'\sim P, a'\sim\pi}\left[R(s,a,s') + \gamma\big(Q^{\pi}(s',a') - \alpha\log\big(\pi(\cdot\,|s')\big)\big)\right]$$

که بصورت تقریبی برابر است با:

$$Q^{\pi}(s,a) \approx r + \gamma(Q^{\pi}(s',\tilde{a}') - \alpha\log\big(\pi(\,\tilde{a}'|s')\big), \quad \tilde{a}'\sim\pi(\cdot\,|s')$$

روش آپدیت الگوریتم

تاع هزینه شبکه Q بصورت زیر است:

$$J_Q(\theta) = \mathbb{E}_{s'\sim P, a'\sim\pi}\left[Q^{\pi}(s,a) - \big(r(s_t,a_t) + \gamma\mathbb{E}_{s'\sim P, a'\sim\pi}\big(Q^{\pi}(s',a')\big) - \alpha\log\big(\pi(\,\tilde{a}'|s')\big)\big)\right]$$

تابع هزینه شبکه سیاست نیز بصورت زیر است:

$$J_{\pi}(\phi) = \mathbb{E}_{s_t\sim D}\left[\alpha\log\big(\pi(\,\tilde{a}'|s')\big) - Q_{\theta}(s_t,a_t)\right]$$

که با روش گرادیان کاهشی توابع هزینه بهینه می‌شوند. و توابع Q نیز بصورت نرم با پارامتر τ آپدیت می‌شوند. ضریب α نیز بصورت زیر آپدیت می‌شود.

$$\mathcal{L}(\alpha) = \alpha(-\log\pi(a|s) - H_{target})$$

$$\nabla_{\alpha}\mathcal{L}(\alpha) = -\log\pi(a|s) - H_{target}$$

که در آن H_{target} یک سطح است که یک تعادل میان exploration-exploitation برقرار می‌کند. معمولا مقدار آن را منفی بُعد فضای عمل می‌گذارند. اگر میزان بی‌نظمی سیاست از آن حد بیشتر بود. عبارت α کاهش یافته و وزن آنتروپی را در تابع هزینه شبکه سیاست کمتر می‌کند. و عامل کاوش کمتری در محیط می‌کند. همچنین اگر بی‌نظمی سیاست باعث افت عملکرد عامل شود؛ در تابع هزینه شبکه سیاست

$Q_\theta(s_t, a_t)$ به سبب گیر افتادن در بهینه محلی کاهش یابد. و حریصانه در آن بهینه محلی بهره‌بـرداری کنـد. این اتفاق به معنی کاهش آنتروپی سیاست است. و باعث افزایش ضریب آلفا می‌شود. تا عامل را از بهینه محلی خارج کند. اما اگر این بهینه سراسری باشد. مقدار q در آن حالت-عمل در بیشترین مقدار ممکن است. و تـابع هزینه سیاست اجازه افزایش آنتروپی سیاست را (که باعث بی‌نظمی و خارج شدن عامل از بیهنـه محلـی مـی-شود) نمی‌دهد.

Algorithm 1 Soft Actor-Critic

Input: initial policy parameters θ, Q-function parameters ϕ_1, ϕ_2, empty replay buffer $\mathcal{D}$
Set target parameters equal to main parameters $\phi_{\text{targ},1} \leftarrow \phi_1$, $\phi_{\text{targ},2} \leftarrow \phi_2$
repeat
  Observe state s and select action $a \sim \pi_\theta(\cdot|s)$
  Execute a in the environment
  Observe next state s', reward r, and done signal d to indicate whether s' is terminal
  Store (s, a, r, s', d) in replay buffer $\mathcal{D}$
  If s' is terminal, reset environment state.
  if it's time to update **then**
    for j in range(however many updates) **do**
      Randomly sample a batch of transitions, $B = \{(s, a, r, s', d)\}$ from $\mathcal{D}$
      Compute targets for the Q functions:

$$y(r, s', d) = r + \gamma(1 - d)\left(\min_{i=1,2} Q_{\phi_{\text{targ},i}}(s', \tilde{a}') - \alpha \log \pi_\theta(\tilde{a}'|s')\right), \quad \tilde{a}' \sim \pi_\theta(\cdot|s')$$

      Update Q-functions by one step of gradient descent using

$$\nabla_{\phi_i} \frac{1}{|B|} \sum_{(s,a,r,s',d) \in B} \left(Q_{\phi_i}(s, a) - y(r, s', d)\right)^2 \qquad \text{for } i = 1, 2$$

      Update policy by one step of gradient ascent using

$$\nabla_\theta \frac{1}{|B|} \sum_{s \in B} \left(\min_{i=1,2} Q_{\phi_i}(s, \tilde{a}_\theta(s)) - \alpha \log \pi_\theta\left(\tilde{a}_\theta(s)|\, s\right)\right),$$

      where $\tilde{a}_\theta(s)$ is a sample from $\pi_\theta(\cdot|s)$ which is differentiable wrt θ via the reparametrization trick.
      Update target networks with

$$\phi_{\text{targ},i} \leftarrow \rho \phi_{\text{targ},i} + (1 - \rho)\phi_i \qquad \text{for } i = 1, 2$$

    end for
  end if
until convergence

شکل ۲ شبکه کد الگوریتم SAC (ورژن اولیه

۳-۲- الگوریتم Conservative Q-Learning

مقدمه

در یادگیری تقویتی آفلاین برخلاف یادگیری تقویتی آنلاین، عامل تنها به یک دیتاست از قبل تهیه شده دسترسی دارد. تعامل نداشتن عامل با محیط باعث می‌شود که Q-value هایی که در دیتاست موجود نیستند ارزش بیش از مقدار واقعی خود را پیدا کنند. این مسئله باعث کاهش عملکرد مدل می‌شود. روش CQL تخمین های بیش از حد خوشبینانه (overestimation) را از بین می‌برد. این روش تنها با اضافه کردن یک عبارت محافظه کارانه به تابع هدف شبکه Q این کار را انجام می‌دهد و قابلیت استفاده با هر الگوریتمی را دارا است. این روش عملکرد فوق العاده ای را در حد مسائل offline RL از خود نشان داده‌است. اما یک عیب آن حساسیت به هایپرپارامتر های مدل می‌باشد.

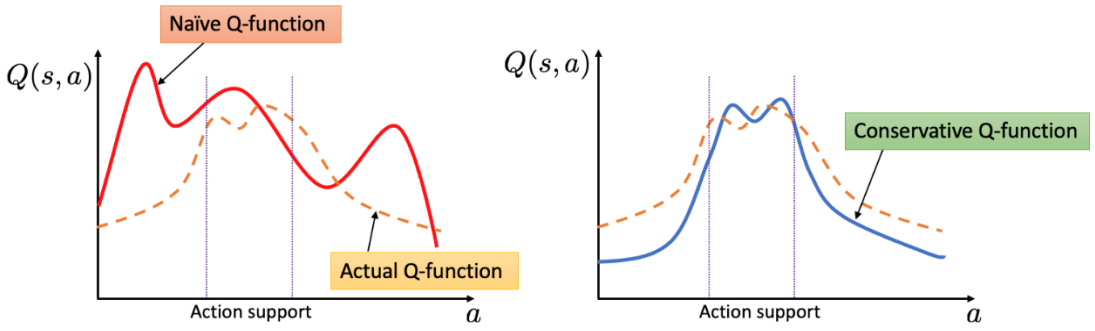
عبارت محافظه کارانه

هدف از اضافه کردن عبارت محافظه کارانه این است که عامل مقدار Q حالت-عمل هایی که بیشتر در دیتاست حضور دارند را افزایش دهد. و عامل مقدار Q حالت-عمل هایی که کمتر در دیتاست حضور دارند را کاهش دهد. تا تخمین های خوش‌بینانه را از بین ببرد. بصورت دقیق‌تر:

$$\mathscr{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}\left[\left(r + \gamma \max_{a'} Q_\theta(s', a') - Q_\theta(s,a)\right)^2\right]$$

$$\mathscr{L}_{CQL}(\theta) = \mathscr{L}(\theta) + \alpha(\mathbb{E}_{(s,a)\sim D}[Q_\theta(s,a)] - \mathbb{E}_{s\sim D, a\sim\mu}[Q_\theta(s,a)]$$

که در آن ضریب آلفا اهمیت مقدار محافظه کارانه را مشخص می‌کند. و μ نیز یک توزیع یکنواخت روی فضای عمل است.



شکل ۳ شهود روش CQL در کم کردن مقدار Q-value ها

۲-۴- روش Behavior Cloning

یکی از ساده‌ترین روش‌های یادگیری تقلیدی[1] روش behavior cloning است. در این روش سعی می‌شود در ازای داشتن یک دیتاست از دوتایی‌های حالت-عمل یک عامل، یک سیاست مشابه سیاست همان عامل ساخت. یک راه ساده برای این کار، آموزش یک شبکه عصبی است بگونه‌ای که بتواند یک نگاشت از حالت‌ها به عمل‌ها انجام دهد. این روش در عین سادگی، توانایی خوبی در تعمیم‌پذیری[2] ندارد.

فصل ۳

فصل ۳: ساختار روش پیشنهادی

با ارائه پیش‌نیاز های روش در فصل پیش، اکنون می‌توان ساختار روش را تشریح کرد.

۳-۱- مقدمه

در این روش فرض می‌شود عامل در یک محیط اجتماعی به همراه بقیه عامل‌ها قرار گرفته است. عامل توانایی دیدن حالت و عمل های بقیه عوامل اجتماعی(در ادامه برای اختصار به عامل اجتماعی SA گفته می-شود.) را دارد. اما توانایی دیدن پاداش را ندارد. سعی عامل بر این است که علاوه بر تعامل با محیط، با دیدن نحوه رفتار SAها بتواند دانشی را استخراج کند که به یادگیری سریعتر و بهتر خودش کمک کند. فرض می-شود SAها تمایلی به آموزش به عامل ندارند و با همدیگر نیز تعاملی ندارند. مهمترین فرض روش ما این است که محیطی که عامل در آن قرار دارد. محیط پیچیده‌ای بوده و علاوه بر آن کاوش در آن هزینه زمانی و مالی

[1] Imitation Learning

[2] Generalization

زیادی دارد. و هزینه محاسبات سخت‌افزاری نسبت به آن قابل صرف نظر است.

۳-۲- مدل‌سازی عامل‌های اجتماعی

برای مدل‌سازی SAها از روش behavior cloning استفاده می‌کنیم. این کار به دو صورت افلاین و آنلاین انجام می‌شود. در روش آفلاین فرض می‌شود که از قبل به تعداد کافی از SAها تجربه وجود دارد. پس در ادامه از خود مدل‌ها استفاده می‌شود.

در روش آنلاین فرض می‌شود SAها به همراه عامل ما در محیط شروع به تعامل کرده و تجربیات به تدریج تولید می‌شود. برای این حالت، هر چند قسمت یکبار روش behavior cloning اجرا شده و مدل ساخته شده را بهبود می‌بخشد.

۳-۳- کارکرد مدل‌ها در روش

در این روش تمامی مدل‌ها به علاوه عامل خودی، یک دسته را تشکل می‌دهند. هر کدام از اعضای این دسته یک سیاست جداگانه‌ای دارد که مدلی از عوامل اجتماعی حاضر در محیط هستند. در هر قسمت، بر اساس یک معیاری، یک عضو ازین دسته انتخاب شده و آن عضو در محیط تعامل می‌کند. معیار انتخاب عامل، باید ارزش بهره‌برداری از سیاست آن عامل را نشان دهد.

۳-۴- معیار امتیازدهی

برای آنکه مشخص کنیم هر عامل چقدر ارزش بهره‌برداری دارد؛ به هر عامل(از جمله خودمان) یک امتیاز می‌دهیم. این امتیاز میانگین نرمال شده پاداش چند قسمت قبلی بوده است که در آن قسمت، آن عامل بازی کرده است. سپس امتیازها را از تابع SoftMax عبور داده، تا به احتمال تبدیل شوند. سپس در ابتدای هر قسمت، با احتمالات گفته شده یک تابع توزیع گسسته می‌سازیم. و از آن نمونه می‌گیریم. نمونه انتخابی عاملی می‌شود که باید در این قسمت بازی کند.

۳-۵- الگوریتم یادگیری عامل

صرف نظر از هر عاملی که درون دسته برای تعامل در محیط در یک قسمت انتخاب می‌شود. تجربیات آن عامل $< s, a, r, s' >$ به حافظه ذخیره‌سازی عامل خودی اضافه می‌شود. بدین ترتیب، این حافظه شامل دیتاهای آنلاین(که توست سیاست عامل تولید شده) و دیتاهای آفلاین (که توسط سیاست بقیه عامل ها تولید شده). می‌شود. الگوریتمی که برای آموزش عامل خودی با استفاده از حافظه استفاده می‌شود الگوریتم Soft Actor-Critic(SAC) می‌باشد. این الگوریتم نقطه ضعف‌های جدی در آموزش با دیتای آفلاین دارد. که باعث می‌شود که شبکه سیاست به خوبی رفتار نکرده و الگوریتم به دلایل زیر شکست بخورد.

- در هنگام آموزش با دیتای آفلاین ممکن است توزیع دیتای آفلاین با آنچه که عامل در محیط می‌بیند فرق کند. به این پدیده Distributional Shift می‌گویند.
- مشکل تخمین بیش از حد نیز در شبکه q وجود دارد. که باعث آموزش یک سیاست غلط می‌شود. به همین سبب، به شبکه critic این الگوریتم یک عبارت محافظه کارانه اضافه می‌شود(که در فصل ۲ بدان پرداختیم) این عبارت جلوی تخمین بیش از حد مقدارهای q را می‌گیرد. و اجازه بهره‌برداری از تجربیات سایر عامل ها را فراهم می‌سازد.
- ضریب α که در الگوریتم SAC وجود دارد خواهان افزایش آنتروپی سیاست است. اما عملکرد بد عامل و کاهش مقدار q جلودار آن است. در نبود دیتای آنلاین هیچ‌وقت عملکرد بد این سیاست مشخص نمی‌شود. و ضریب α واگرا می‌شود.

برای حل مشکلات بالا، الگوریتم CQL معرفی شده است که در فصل ۲ بدان پرداختیم. این الگوریتم با حل مشکل Distributional Shift و overestimation باعث تخمین درست مقدارهای q می‌شود. و سیاست نیز حتی در صورت ندیدن محیط واقعی با فیدبک‌هایی که از شبکه Q می‌گیرد؛ خود را اصلاح می‌کند. و ضریب α نیز واگرا نمی‌شود.

۳-۶- مزایای مدل

این روش دو مزیت عمده دارد. مزیت اول روش این است که در همان چند قسمت اول، عامل های خبره امتیاز بالا و عامل های ضعیف امتیاز پایین می‌گیرند. و عامل های خبره شانس بیشتری برای تعامل با

محیط پیدا می‌کنند. این پدیده همزمان که باعث می‌شود برای آموزش شبکه خودی بصورت آفلاین expert trajectory تهیه شود، مادامی که شبکه خودی مشغول آموزش است به کمک الگوریتم SAC, CQL عملکرد خود را بصورت ترکیبی آنلاین و آفلاین بهبود می‌بخشد. و تا زمانی که به حد خبرگی برسد. و بقیه عامل‌های داخل دسته را کنار بزند. تقریبا همواره امتیاز بالایی در محیط به نام او ثبت می‌شود.

فصل ۴

فصل ۴: پیاده‌سازی و ارزیابی نتایج

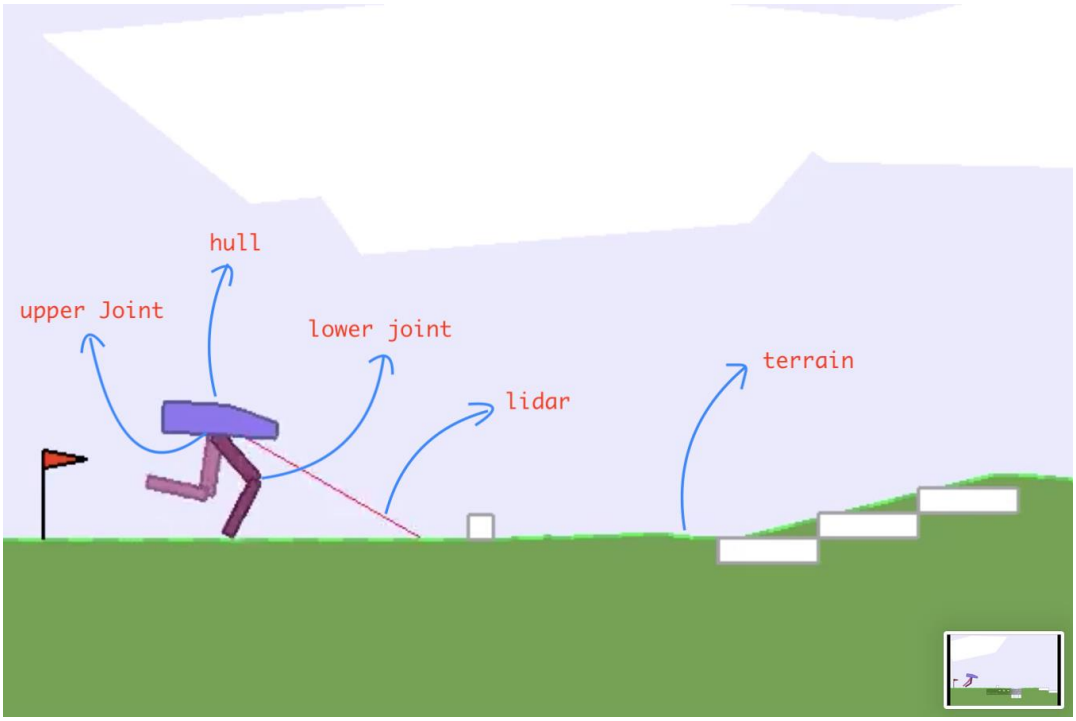
پس از شرح الگوریتم پیشنهاد شده در فصل قبل، در این فصل به پیاده‌سازی این روش و ارزیابی نتایج می‌پردازیم:

۴-۱- مقدمه

همانطور که در مقدمه گفته شد، هدف از این الگوریتم، سرعت بخشیدن به یادگیری در محیط‌هایی است که پیچیدگی زیادی دارند و یا تعامل با آن‌ها هزینه‌بر است. به همین دلیل محیط انتخابی نیز باید چنین خصوصیاتی داشته باشد.

۴-۲- محیط پیاده‌سازی

برای شبیه‌سازی الگوریتم از محیط BipedalWalker کتابخانه Gymnasium که کتابخانه مخصوص یادگیری تقویتی است استفاده می‌کنیم. در این محیط یک موجود ۲ پا که در هر پا یک زانو دارد(مشابه انسان). باید یاد بگیرد که در یک سطح صاف راه برود. در حالت Hardcore(دشوار) محیط دارای مانع، شیب و چاله بوده که یادگیری را چالشی می‌کند. در این پروژه از نسخه Hardcore استفاده شده.



شکل ۴ نمای محیط شبیه‌سازی

به ازای هر مقداری که عامل در محیط حرکت کند پاداش کمی می‌گیرد. در صورت برخورد به زمین امتیاز منفی ۱۰۰ دریافت می‌کند. فضای عمل ۴ مفصل پا هستند. و فضای حالت شامل سرعت، موقعیت، شتاب و شتاب زاویه ای پاها به علاوه ۱۰ لیزر فاصله‌سنج می‌باشد. که مجموعا ۲۴ بعد را می‌سازند. حداکثر پاداش این محیط ۲۳۰ و حداقل آن ۱۵۰- است.

۳-۴- عامل شخصی

عامل شخصی برای یادگیری از الگوریتم SAC+CQL استفاده می‌کند. شبکه‌های مورد استفاده fully connected با ۳ لایه پنهان با ۲۵۶ نورون می‌باشد. سایر پارامترها برای هر آزمایش fine tune شدند و مقدار ثابتی ندارند.

۴-۴- عامل‌های اجتماعی

برای عامل‌های اجتماعی، انواع حالات را در نظر می‌گیریم. یک عامل خبره محیط Hardcore، عامل بعدی نیمه خبره محیط Hardcore، عامل بعدی خبره محیط عادی و یک عامل تصادفی. جزئیات آزمایش‌ها بشکل زیر است. امتیاز عامل خبره Hardcore برابر ۱۹۰ و امتیاز عامل نیمه خبره Hardcore برابر ۸۰ می‌باشد.

جدول 1 جدول جزئیات شبیه‌سازی‌ها

شماره	عامل‌های حاضر	Mode
1	تکی	-
2	خبره دشوار، خبره عادی، تصادفی	Offline
3	خبره دشوار، خبره عادی، تصادفی	Online
4	نیمه خبره دشوار	Offline
5	خبره عادی، تصادفی	Offline

برای حالت آفلاین فرض می‌کنیم که مدل کاملی از سیاست عامل‌های اجتماعی داریم. در حالت آنلاین بقیه SAها همزمان با عامل‌خودی در محیط زندگی می‌کنند. برای مدل سازی سیاست

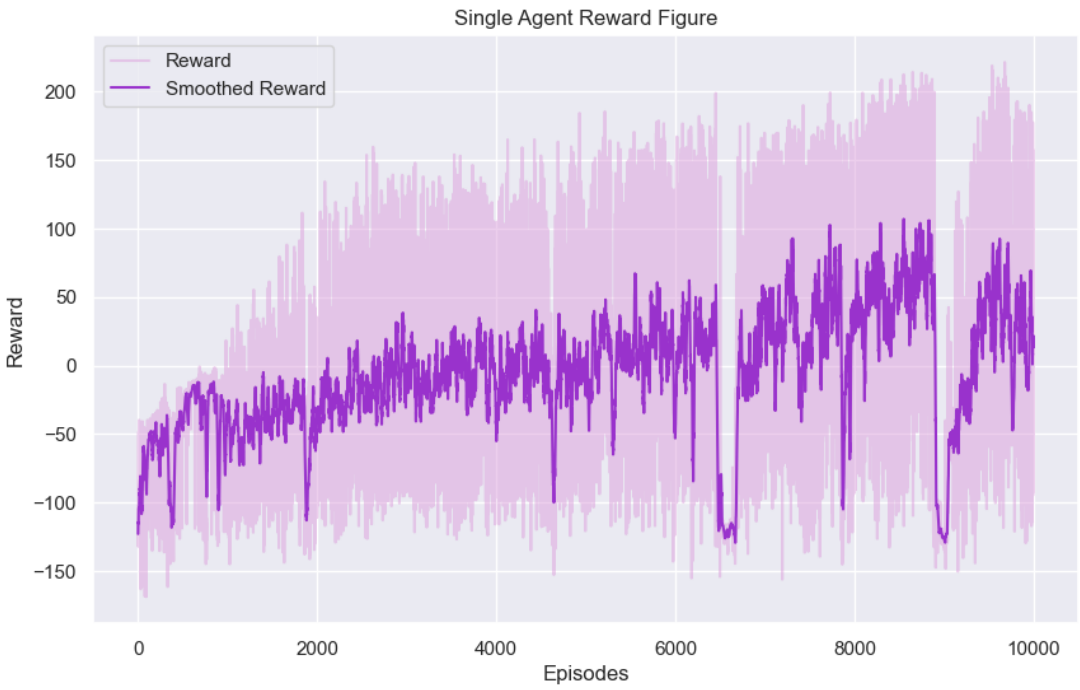
SAها از روش Behavior Cloning در قسمت های ۳۵۰،۲۰۰،۱۰۰،۵۰،۲۰،۱۰ محیط استفاده می‌کنیم. در قسمت ۳۵۰ مدل بدست آمده تفاوت اندکی با عامل واقعی دارد.

۵-۴- نتایج

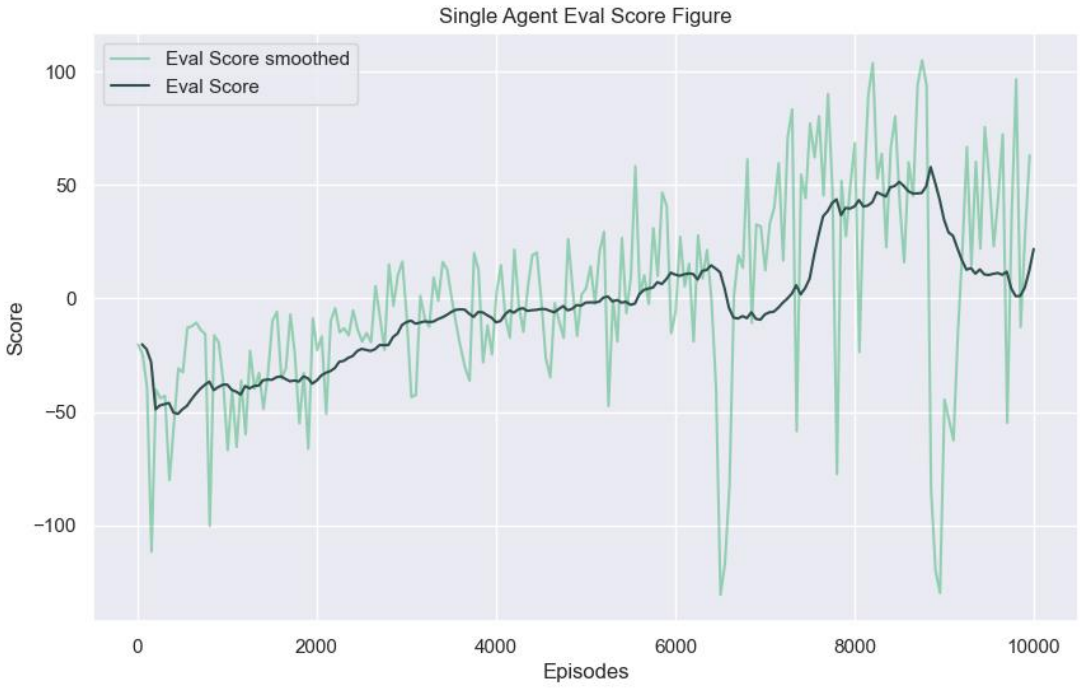
آزمایش ۱:

آزمایش ۱ برای مقایسه کارکرد حالت تکی با حالت اجتماعی است.

همانطور که در شکل مشخص است، در حالت تکی، عامل باید زمان زیادی را صرف یادگیری کند. و با تجربه ۱۰هزار قسمت توانسته به میانگین پاداش ۵۰ برسد.



شکل ۵ نمودار پاداش عامل در طول آموزش



شکل ۶ نمودار پاداش عامل در حالت ارزیابی

به دلیل آزادی محیط در قرار دادن مانع‌ها، سختی محیط ها بصورت رندوم در هر قسمت تغییر می‌کند. به همین دلیل واریانس پاداش‌ها برای عامل تکی و نیز سایر عامل‌ها بالا است.

آزمایش ۲

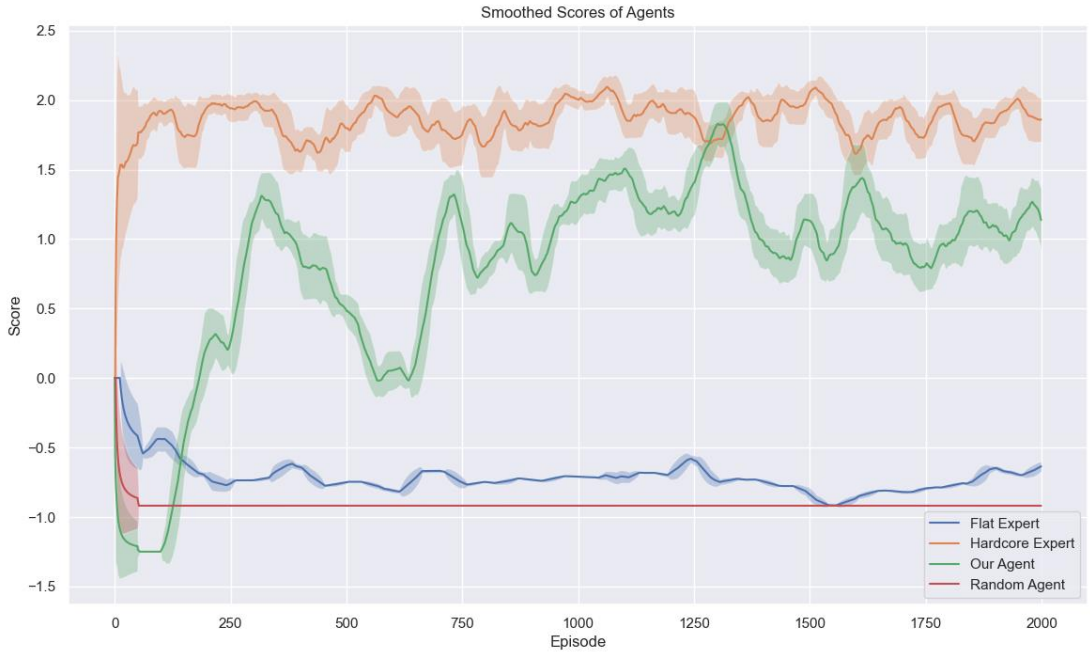
در آزمایش ۲ می‌بینیم که امتیاز عامل ها بصورتی که پیش‌بینی می‌شد درآمد. عامل خبره دشوار توانست امتیاز حدود ۱.۸۸ را کسب کند در حالی که عامل خبره عادی به دلیل نا آشنایی با محیط دشوار، در مقابله با اولین مانع به زمین می‌افتد. و امتیاز حدود ۰.۷- را کسب می‌کند. عامل رندوم نیز عدد ثابت ۰.۹۳- را کسب می‌کند.

در این آزمایش می‌بینیم که الگوریتم SAC+CQL به خوبی توانست است دیتای آفلاین و آنلاین را ترکیب کرده و عملکرد عامل را در زمان کوتاه کمتر از ۵۰۰ قسمت افزایش دهد.

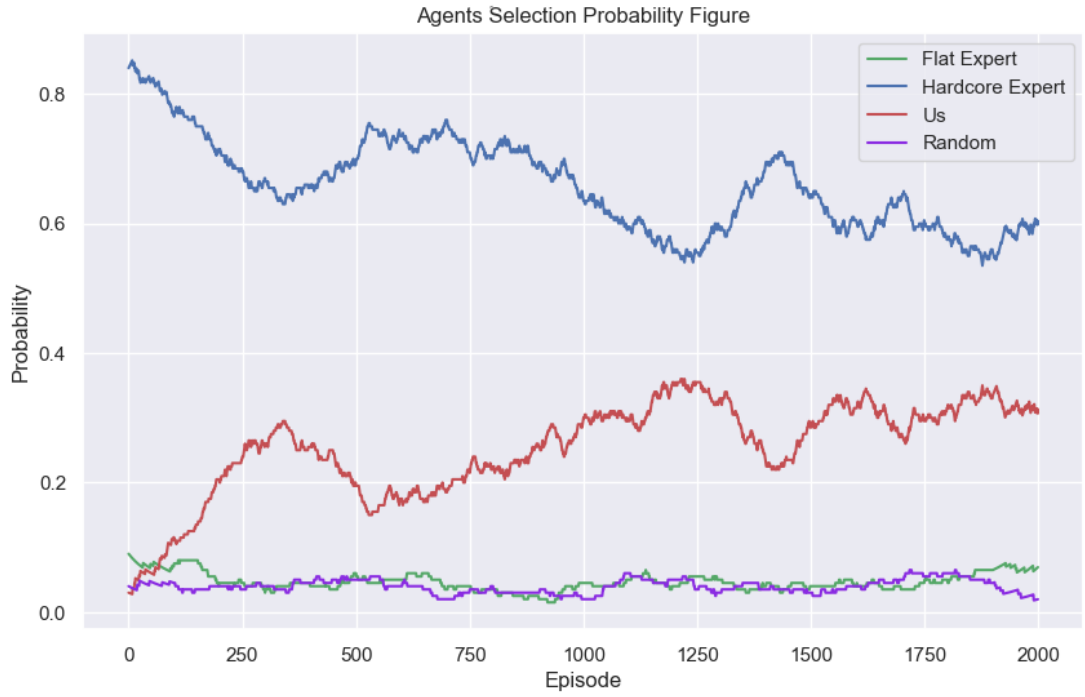
همچنین دسته‌ای که شامل مدل‌سازی عامل‌ها است. نیز به خوبی کار کرده و توانسته دفعات زیادی را به عامل خبره دشوار، و دفعات نادری را به دو عامل اجتماعی دیگر اختصاص دهد. عامل خودی نیز همگام با یادگیری و بهبود عملکرد خود، کم کم خود را نزدیک عامل خبره دشوار رسانده و با کسب امتیاز ۱۵۰، تعداد قابل توجهی از قسمت‌ها را در دست خود می‌گیرد.

شکل ۷ نمودار پاداش دسته در طول آموزش

شکل ۸ نمودار پاداش عامل در حالت ارزیابی



شکل ۹ نمودار امتیاز مدل‌ها در طول آموزش



شکل ۱۰ نمودار احتمال انتخاب مدل‌ها در طول آموزش

در این آزمایش، اثر حضور نویز نیز بررسی شد. که برای طولانی نشدن فصل فقط نتایج بیـان مـی-شود.

- با اعمال نویز ۱۰٪ دامنه فضای عمل، روش توانایی یادگیری ندارد. و شکست می‌خورد
- با اعمال نویز ۵٪ دامنه فضای عمل، روش دچار حدود ۵۰ واحد افت امتیاز می‌شود.
- با اعمال نویز ۱٪ دامنه فضای عمل، اثر اندکی در روش مشاهده می‌شود.

آزمایش ۳

در این آزمایش دیگر مدل کاملی از عامل‌ها در دسترس نداریم. به همین علت، با شروع تعامل عامل خودی در محیط، عامل‌های دیگر نیز شروع به تعامل کرده و آرام آرام تجربیات آنان در اختیار ما قرار می‌گیرد.

روش ایجاد مدل از عامل‌ها بدین صورت است که در بازه های زمانی مشخص تمامی داده‌های که از عامل بدست آماده جمع می‌شوند. و به کمک روش Behavior Cloning مدل آموزش داده می‌شود. در تمامی طول آموزش، از مدل ساخته شده توسط روش BC بجای مدل اصلی انتخاب می‌شود

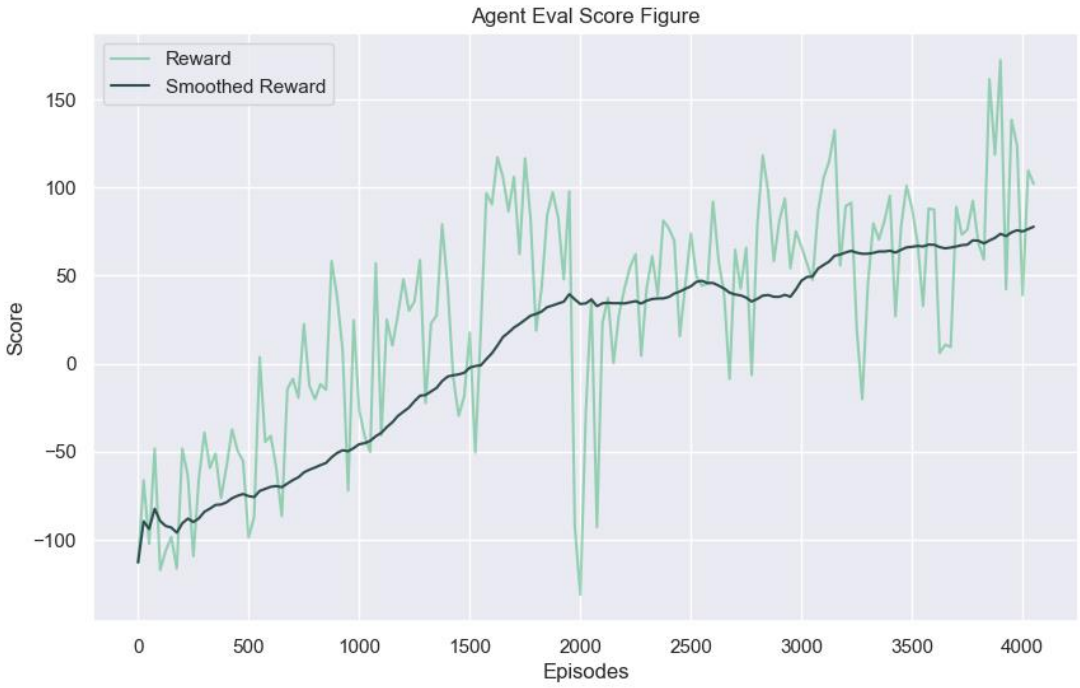
جدول ۲ جدول امتیاز مدل خبره دشوار بر حسب تعداد قسمت‌های در دسترس

امتیاز	قسمت های در دسترس
۵۵	۱۰
۷۶	۲۰
۱۲۴	۵۰
۱۵۱	۱۰۰
۱۶۰	۲۰۰
۱۷۸	۳۵۰

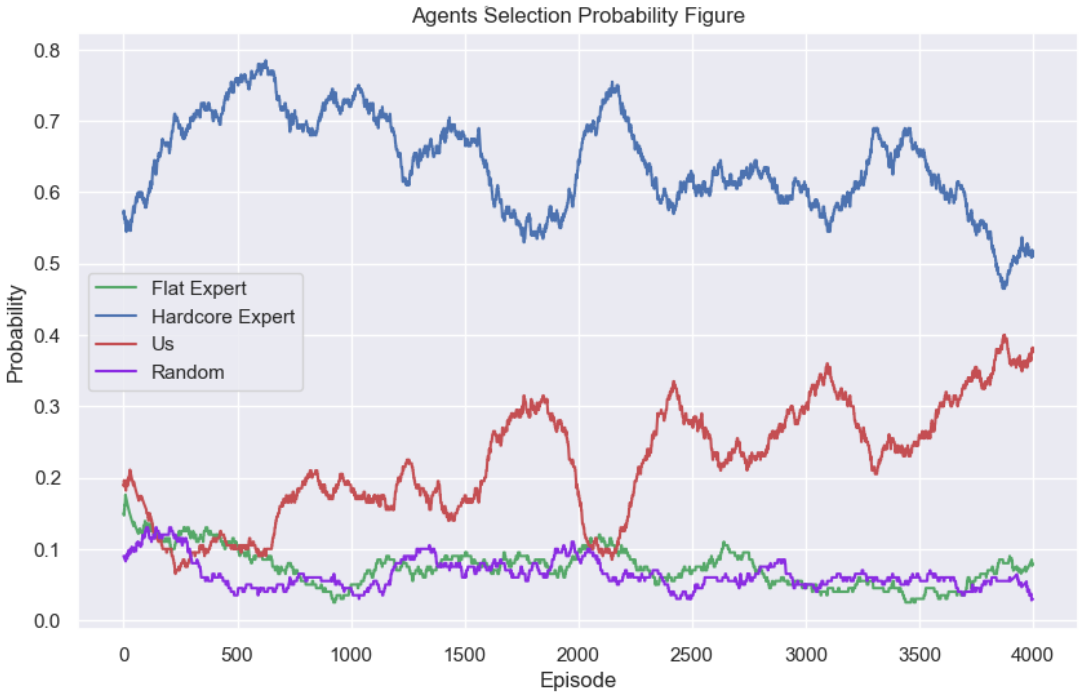
در آزمایش ۳ می‌بینیم که مدل خبره دشوار در حدود قسمت ۳۵۰ به امتیازی نزدیک امتیاز خود این عامل می‌رسد. و ادامه آموزش شباهت به آموزش پیشین دارد. اما به دلیل پایدار نبودن مدل عامل اجتماعی در ابتدای آموزش، این شبیه‌سازی با اغتشاش همراه است. در نهایت دسته توانست به امتیاز ۱۵۰ برد. همچنین عامل خودی توانست امتیاز خود را بسیار نزدیک مدل خبره دشوار کند. و در ارزیابی به امتیاز ۱۶۵ نیز دست یافت. و سهمی حدود ۳۷٪ از انتخاب های دسته را به خود اختصاص داد.

این آزمایش با موفقیت نشان داد که روش پیشنهادی قابلیت پیاده‌سازی آنلاین را نیز دارا می‌باشد که یک نقطه قوت بسیار برجسته به شمار می‌رود.

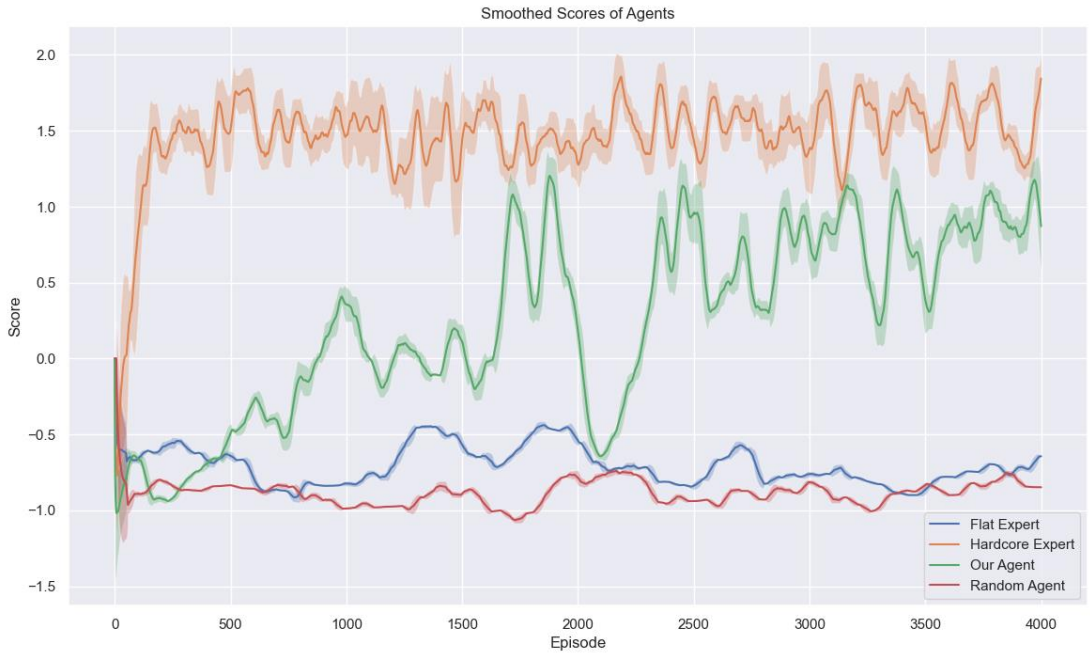
شکل ۱۱ نمودار پاداش دسته در طول آموزش



شکل ۱۲ نمودار پاداش عامل در حالت ارزیابی



شکل ۱۳ نمودار احتمال انتخاب مدل ها در طول آموزش



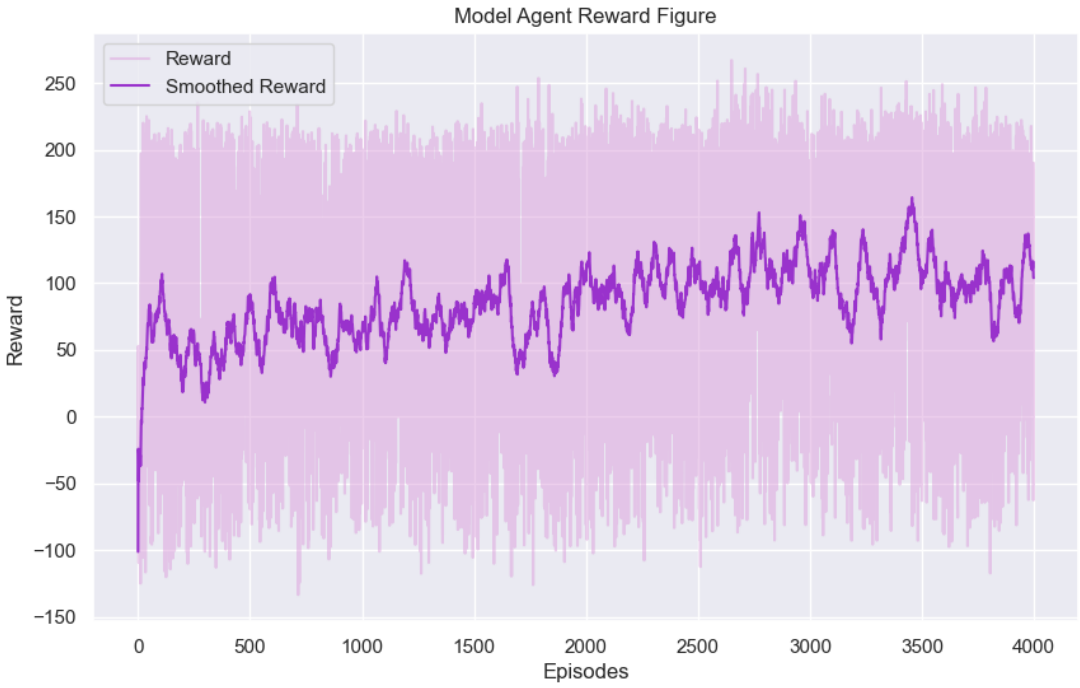
شکل ۱۴ نمودار امتیاز مدل ها در طول آموزش

آزمایش ۴

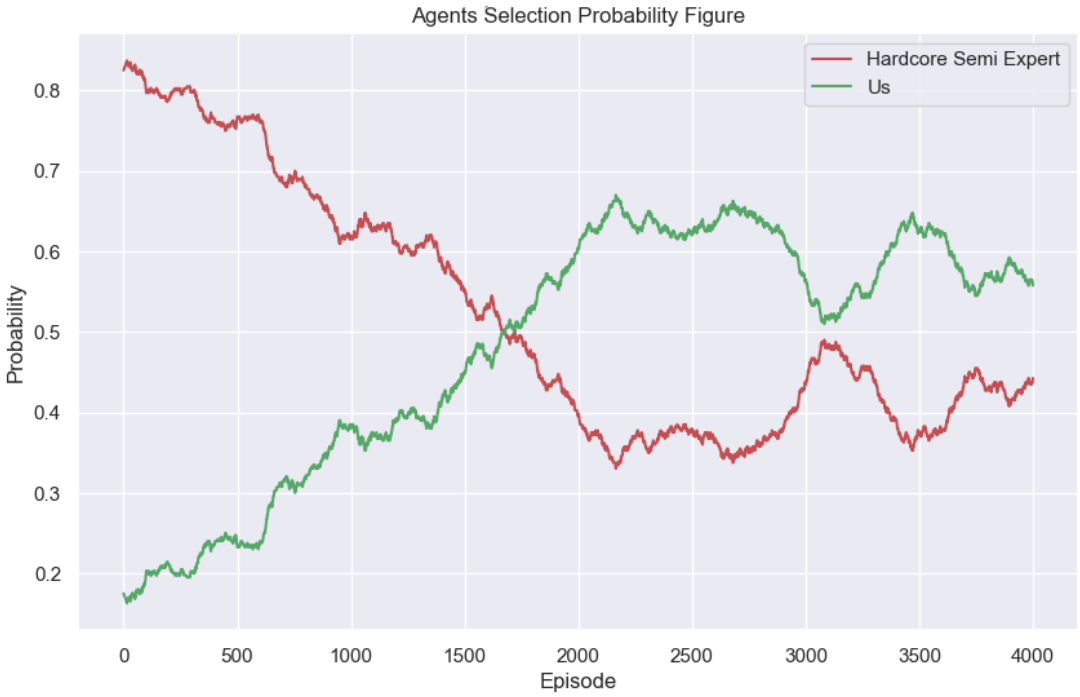
در این آزمایش عامل در کنار یک عامل نیمه خبره قرار می‌گیرد. هدف ازین آزمایش بررسی این موضوع است که آیا عامل توانایی بهره برداری از عامل‌های غیر خبره را نیز دارد یا خیر.

با نگاهی به نمودار ها می‌توان دید که عامل به خوبی توانسته از تجربه‌های آفلاین بهره برداری کند و در نیمه‌ی آموزش موفق به پیشی گرفتن از عامل نیمه خبره در امتیاز و احتمال انتخاب می‌شود.

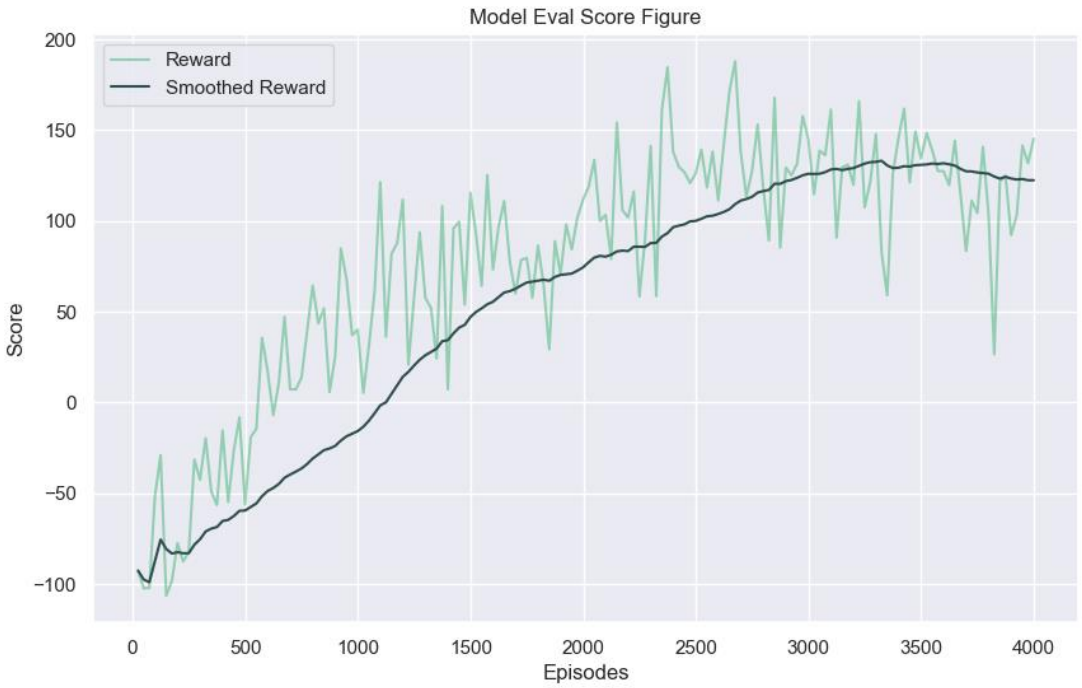
اما همانطور که انتظار می‌رود. در مقایسه با آزمایش ۲، عملکرد دسته حدود ۵۰ واحد کمتر از آزمایش ۲ می‌باشد. همچنین عامل خودی برای رسیدن به امتیاز حدود ۱۵۰ به ۴۰۰۰ قسمت نیاز دارد در حالی که در آزمایش ۲ به فقط ۲۰۰۰ قسمت نیاز بود.



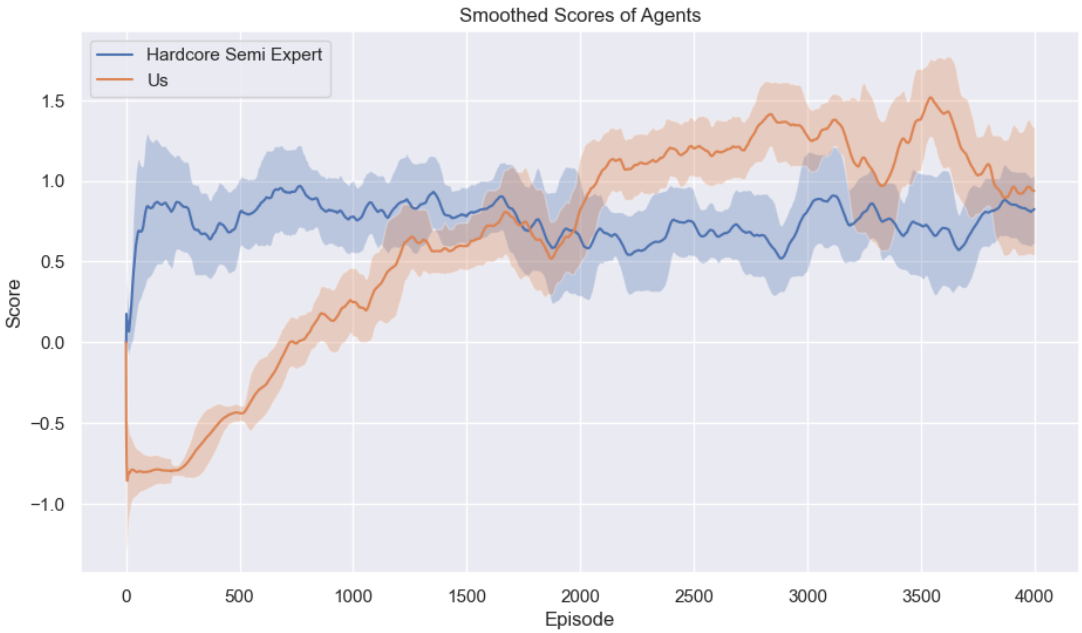
شکل ۱۵ نمودار پاداش دسته در طول یادگیری



شکل ۱۶ نمودار احتمال انتخاب مدل‌ها در طول یادگیری



شکل ۱۷ نمودار پاداش عامل در حالت ارزیابی



شکل ۱۸ نمودار امتیاز مدل‌ها در طول یادگیری

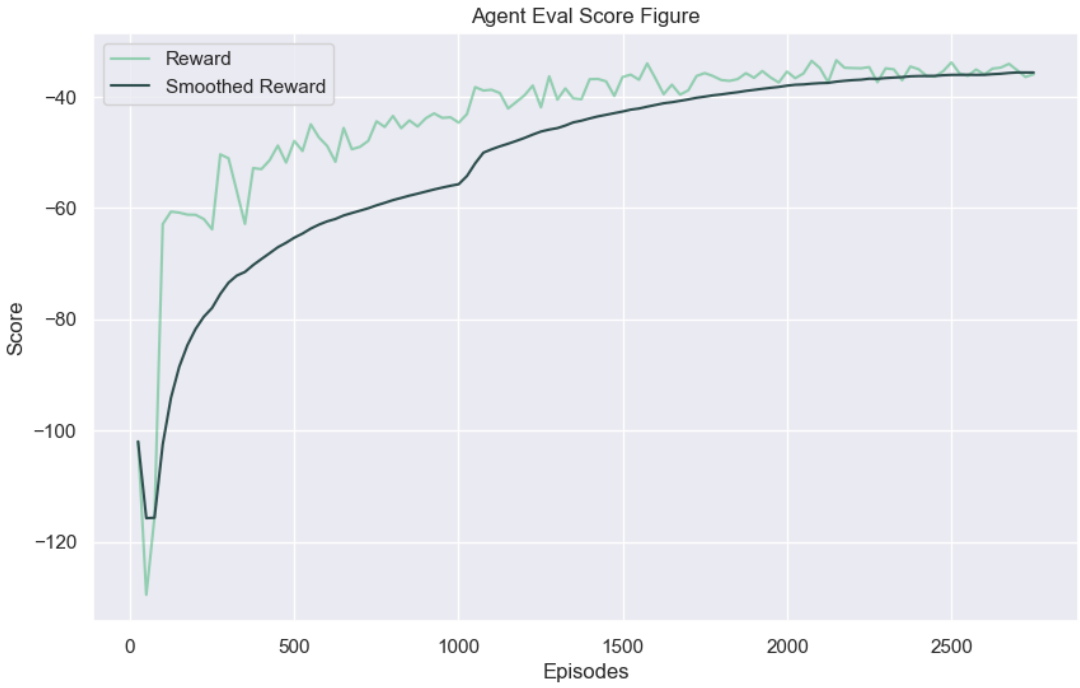
آزمایش ۵

در این آزمایش هیچ عامل خبره‌ای وجود ندارد. یک عامل تصادفی و عامل دیگر در این محیط بدون تجربه است.

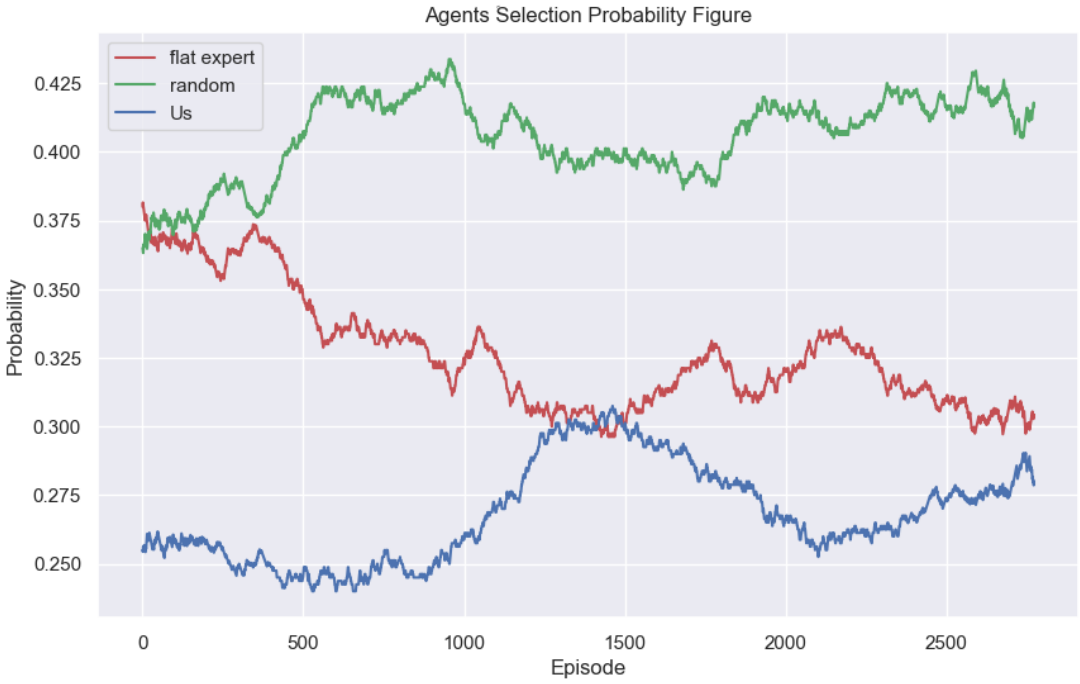
نتایج نشان می‌دهد. الگوریتم در این محیط توانایی بهره‌برداری از تجربیات عامل‌های اجتماعی را ندارد و الگوریتم شکست می‌خورد.

البته در مقاله روش CQL اشاره شده که این روش توانایی بهره‌برداری از دیتاست تصادفی را دارا می‌باشد. اما با توجه به تعداد هایپرپارامترهای این الگوریتم نیاز به fine-tunning بسیار برای جواب گرفتن دارد. و سنگین بودن شبیه‌سازی روش، این مهم را امکان‌پذیر نمی‌کند.

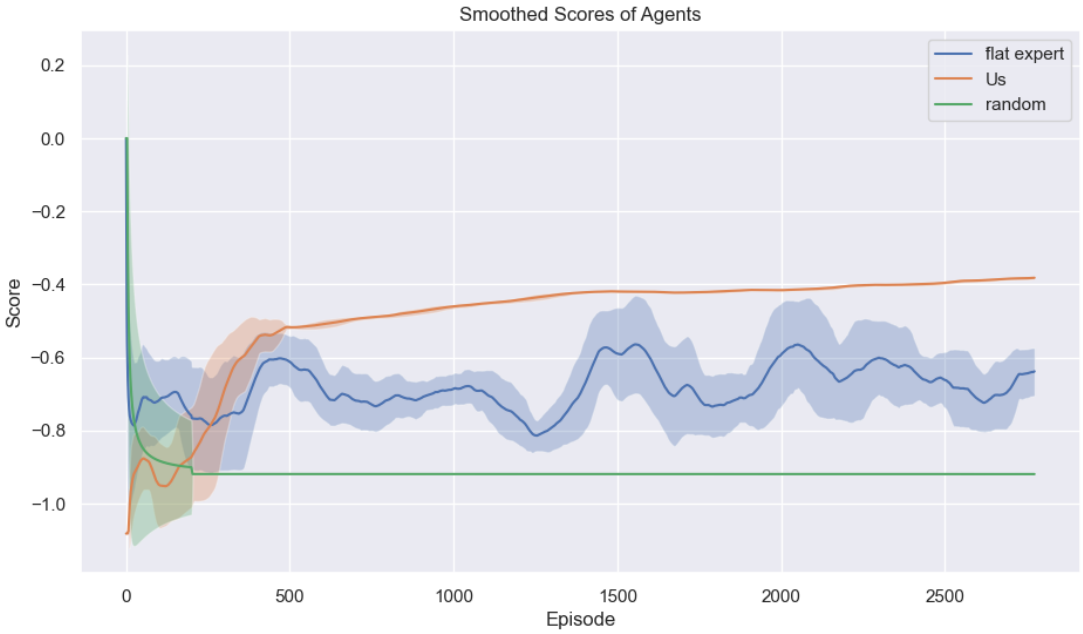
شکل ۱۹ نمودار پاداش دسته در طول آموزش



شکل ۲۰ نمودار پاداش عامل در حالت ارزیابی



شکل ۲۱ نمودار احتمال انتخاب مدل‌ها در طول ارزیابی



شکل ۲۲نمودار امتیاز مدل‌ها در طول آموزش

۴-۶- خلاصه و جمع‌بندی

با انجام آزمایشات بالا مشخص شد که روش به خوبی کار می‌کند و در ۳ از ۴ حالت موفق به یادگیری موثر شده است. به عبارتی در صورت وجود یک عامل خبره یا نیمه خبره در محیط، روش توانایی بالایی در شناسایی و بهره‌برداری از تجربیات آن عامل دارد. اما در صورت نبود هیچ عامل خبره‌ای روش نمی‌تواند به تنهایی یاد بگیرد. همچنین در کنار عامل‌های خبره و نیمه‌خبره، دسته از همان ابتدا پاداشی به اندازه عملکرد بهترین عامل موجود در محیط را به نام خود ثبت می‌کند. همچنین یک چالش اساسی در زمینه شبیه‌سازی، این است که الگوریتم CQL برای دیتاست‌های آفلاین طراحی شده است. در حالی که بعد از مدتی که عامل با تجربه شده و مقداری از سهم تعامل با محیط را به خود اختصاص می‌دهد. دیتاست ترکیبی از دیتاهای آفلاین و آنلاین می‌شود. که همواره در حال تغییر است. و حد پایینی که این روش در محاسبه مقدارهای q می‌گیرد باعث ایجاد خطا شود. به همین دلیل است که روش CQL به هایپرپارامتر ها حساس است. بطوری که نیاز به چندین بار شبیه‌سازی است تا نتایج رضایت بخش شود.

فصل ۵

فصل ۵: جمع‌بندی، نتیجه‌گیری و پیشنهادها

۵-۱- نتیجه‌گیری

بصورت کلی عملکرد روش پیشنهادی بسیار رضایت بخش بوده و علیرغم سنگین بودن توانایی بالایی در تشخیص سطح خبرگی و استخراج تجربه سایر عامل‌ها دارد. و در صورت وجود چنین عامل‌هایی به سرعت خود را به سطح آن‌ها می‌رساند. اما اگر همه عامل‌های یک محیط بی‌تجربه باشند؛ عامل نمی‌تواند دانشی استخراج کند و یا به حالت یادگیری تکی برگردد. پیاده‌سازی این روش بسیار ساده بوده و الهام گرفته از الگوریتم Policy Gradient است. که برای انتخاب یک مدل صرفا بهترین آن را برنمی‌دارد. بلکه با ایجاد یک تابع توزیع شانس بیشتری به عامل‌های بهتر می‌دهد.

هنگامی که یک عامل وارد محیطی جدید می‌شود. و می‌خواهد به سرعت خود را با آن محیط وفق بدهد. روش پیشنهادی یک گزینه عالی می‌باشد. که بدون طی کردن مسیر طولانی آموزش، تجربیات افراد باتجربه محیط را مستقیما به عامل خودی منتقل می‌کند. مثلا یک ربات که با دیدن نحوه راه رفتن یا نواختن ساز آدم‌ها ، مدل-سازی از مفصل‌ها و ماهیچه‌ها انجام داده و با سرعت زیادی راه رفتن یا مهارت نوازندگی را یاد می‌گیرد. یا یک مدل هوش مصنوعی که وظیفه آموزگاری را بر عهده دارد می‌تواند با زیرنظر داشتن سایر آموزگاران، روش‌ها و شگردهایی که آنها در تدریس بکار می‌برند را شناسایی کرده، و برای خود بکار گیرد.

۵-۱-۱- محدودیتها

دو محدودیت اصلی در این روش وجود دارد:

- سنگین بودن: روش SAC بهترین و سنگین ترین الگوریتم جنرال یادگیری تقویتی می‌باشد. همچنین روش CQL نیز یک سربار دیگر به آن اضافه می‌کند. در روش آنلاین نیز مدل‌سازی از سایر عامل‌ها خود نیازمند زمان قابل توجه است. که آن را برای بسیاری از کاربرد ها نامناسب می‌کند.
- حساسیت روش CQL: این روش بسیار به هایپرپارامتر حساس است. و با هر بار تغییر عامل‌های اجتماعی نیاز به fine-tune کردن بسیار برای پیدا کردن هایپرپارامتر های مناسب است.

۵-۱-۲- پیشنهادها

برای ادامه تحقیقات ۴ مسیر زیر پیشنهاد می‌شود.

- از آنجایی که روش SAC یک روش آنلاین و روش CQL یک روش آفلاین است. می‌توان مموری دسته را دو قسمت کرد و قسمتی که از تعامل عامل خودی بدست آمده را به SAC داد. و قسمت‌هایی که مدل‌های دیگر بازی کرده‌اند. را به CQL داد. و دو روش همزمان مستقل از هم شبکه‌های عامل را آپدیت کنند.
- پیدا کردن یک رابطه بین هایپرپارامترهای روش CQL و سطح خبره بودن دیتای موجود در مموری بطوری که هایپرپارامترها خود بصورت دینامیک آپدیت شوند.
- استفاده از روش‌های دیگر یادگیری آفلاین همانند Implicit Q-Learning در کنار SAC برای بهبود تجربه یادگیری ترکیبی.
- توسعه روش به محیط‌های Partial Observable که در آن دریافت تجربیات دیگران منوط به حضور آن ها در میدان دید عامل خودی است.

فصل۶

فصل ۶: مراجع

مراجع

[1] T. Haarnoja, A. Zhou, P. Abbeel and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, Stockholm, Sweden, 2018.

[2] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft Actor-Critic Algorithms and Applications," arXiv:1812.05905, 2018.

[3] A. Kumar, A. Gupta, and S. Levine, "Conservative Q-Learning for Offline Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[4] S. Zhuang, X. Ma, Y. Li, and H. Liu, "Cal-QL: Calibrated Offline RL Pre-Training for Efficient Online Fine-Tuning," *arXiv preprint arXiv:2301.12345*, 2023. [Online]. Available: https://arxiv.org/abs/2301.12345

پیوست‌ها

Abstract:

Training a reinforcement learning agent has always been a challenging and time-consuming process. Many environments are high-dimensional or real-world systems, where interactions occur very slowly due to the lack of modeling capabilities. In these systems, the presence of other agents offers us an excellent opportunity to extract their experience and knowledge to enhance the speed and quality of learning. However, other agents may differ in their level of expertise and objective functions. More importantly, they may be reluctant to share their policies and rewards.

To address this issue, we have proposed a new method that gradually builds a model of all the agents present in the environment. Then, it assigns a score to each agent based on the performance of their model. Agents with higher scores are given more opportunities to interact with the environment, providing our agent with high-quality expert trajectories to accelerate learning.

Experimental results of this method show that it not only speeds up the agent's learning process by several folds but also consistently achieves high rewards even in the early stages of training, due to the use of the expert agent's model policy instead of its own policy.

Keywords:

Reinforcement Learning, Social Agent, Behavior Cloning, Soft Actor-Critic, Conservative Q-Leaning

University of Tehran

College of Engineering

School of Electrical and Computer Engineering

Improving Learning of Social Learning Agent by Observing the Behavior of Other Agents

A thesis submitted to the Undergraduate Studies Office

In partial fulfillment of the requirements for

The degree of Bachelor in Science in

Electrical Engineering

By:

Nima Zamanpour

Supervisor:

Dr. Majid Nili Ahmadabadi