دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

هوش مصنوعی

پاییز ۱۴۰۲

اساتید: محمدحسین رهبان، مهدیه سلیمانی باغشاه

گردآورندگان: امیرحسین حاجی‌محمدرضایی، آیلین رسته، حسین گلی، محمد لطفی، بنیامین ملکی

تمرین سوم — **فرآیند تصمیم‌گیری مارکوف، یادگیری تقویتی** — مهلت ارسال: ۴ آذر

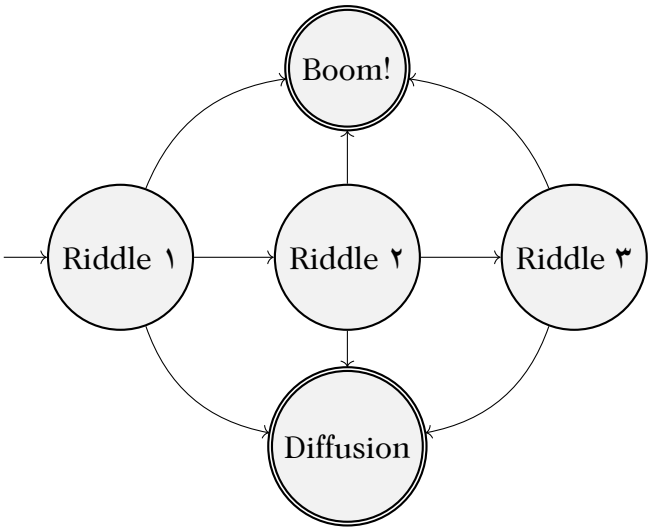
- مهلت ارسال پاسخ تا ساعت ۲۳:۵۹ روز مشخص‌شده است.
- در طول ترم امکان ارسال با تاخیر پاسخ همه‌ی تمارین تا سقف **۳** روز و در مجموع **۵** روز، وجود دارد. پس از گذشت این مدت، پاسخ‌های ارسال‌شده پذیرفته نخواهند بود. همچنین، به ازای هر روز تأخیر غیر مجاز **۲۴** درصد از نمره تمرین به صورت ساعتی کسر خواهد شد. جزئیات نحوه اعمال تاخیرها را می‌توانید در سایت درس مشاهده کنید.
- هم‌کاری و هم‌فکری شما در انجام تمرین مانعی ندارد اما پاسخ ارسالی هر کس حتما باید توسط خود او نوشته شده باشد.
- در صورت هم‌فکری و یا استفاده از هر منابع خارج درسی، نام هم‌فکران و آدرس منابع مورد استفاده برای حل سوال مورد نظر را ذکر کنید.
- لطفا تصویری واضح از پاسخ سوالات نظری بارگذاری کنید. در غیر این صورت پاسخ شما تصحیح نخواهد شد.

سوالات نظری (۱۳۰ نمره)

۱. (۳۰ نمره) خطری جدید شهر گاتهام را تهدید می‌کند! ریدلر (Riddler) در یکی از نقاط پرتردد شهر بمبی را پنهان کرده است و این بمب به زودی منفجر خواهد شد. از طرفی، ریدلر یک سلسله معما برای بتمن طراحی کرده است که با حل کردن هر معما، بخشی از موقعیت مکانی بمب مشخص می‌شود. همچنین، ریدلر مشخص نکرده است که کی قرار است این بمب منفجر شود، بنابرین اگر بتمن بخواهد وقت خود را صرف حل کردن معماهای بیشتر بکند، ممکن است در این میان بمب منفجر شود!

ریدلر سه معما برای بتمن فراهم کرده است. بتمن در هر معما می‌تواند بر اساس اطلاعاتی که تا آنجا از موقعیت مکانی بمب بدست آورده، مکان دقیق آن را «حدس» بزند و به سراغ آن برود. در صورتی که حدسش درست بوده باشد، بمب را خنثی می‌کند و در غیر این صورت، به مکان اشتباهی می‌رود و بمب منفجر خواهد شد. پس از اینکه یک معما را حل کرد، کار دیگری که می‌تواند بکند این است که به سراغ «حل معمای بعدی» برود. در فاصله‌ی زمانی انتقال بین معماها ممکن است که بمب منفجر شود، اما، ممکن است که این اتفاق هم نیفتد و او خودش را با موفقیت به معمای بعد برساند. بتمن با تجزیه و تحلیلی که از شرایط انجام داده و با توجه به شناختی که از ریدلر دارد، توانسته است که وضعیت را با MDP شکل ۱ مدل کند. در این MDP دو اکشن در هر معما قابل انجام است:

- guess (حدس زدن موقعیت بمب و رفتن به آنجا)



شکل ۱: استیت‌ها و جابه‌جایی‌های ممکن میان آن‌ها در MDP

- next (حل کردن معمای بعدی)

بر این اساس، مقادیر تابع T این MDP به صورت زیر هستند:

$$T(Riddle1, next, Riddle2) = 0.5, T(Riddle1, next, Boom!) = 0.5$$
$$T(Riddle2, next, Riddle3) = 0.5, T(Riddle2, next, Boom!) = 0.5$$
$$T(Riddle1, guess, Diffusion) = 0.3, T(Riddle1, guess, Boom!) = 0.7$$
$$T(Riddle2, guess, Diffusion) = 0.4, T(Riddle2, guess, Boom!) = 0.6$$
$$T(Riddle3, guess, Diffusion) = 0.9, T(Riddle3, guess, Boom!) = 0.1$$

همچنین تابع R به صورت زیر تعریف می‌شود:

$$R(*, guess, Diffusion) = 1000, R(*, *, Boom!) = -1000$$
$$R(Riddle1, next, Riddle2) = 150, R(Riddle2, next, Riddle3) = 400$$

از آنجایی که بتمن باید سریعا به محل اولین معما حرکت کند، از شما خواسته در این فاصله زمانی،با مقداردهی اولیه‌ی صفر به تمامی V(s)ها، و اجرای الگوریتم Value Iteration تا ۳ مرتبه، بهترین سیاست را برای بتمن مشخص کنید. در حقیقت بگویید که بهتر است چند معما را سعی کند حل کند و سپس، در کدام معما مکان بمب را حدس بزند. $(\gamma = 0.9)$
لطفا راه حل را به صورت کامل نوشته و به جواب پایانی اکتفا نکنید.

۲. (۲۵ نمره) درستی یا نادرستی عبارات زیر را با استدلال مشخص کنید.

- فرض کنید یک MDP متناهی با صرفا یک وضعیت شروع و چندین وضعیت نهایی (ترمینال) داریم. در صورتی که در شروع الگوریتم Value Iteration تمامی مقادیر $V(s)$ استیت‌های غیرترمینال را صفر در نظر بگیریم و این مقدار را برای استیت‌های ترمینال، ناصفر (و برابر پاداشی که در انتهای بازی دریافت خواهیم کرد) قرار دهیم، حداقل $\mathcal{L}$ مرحله (iteration) طول خواهد کشید تا برای اولین بار مقدار استیت شروع یا $V(s_0)$ مقداری غیرصفر یابد. در جمله قبل، $\mathcal{L}$ برابر با طول کوتاه‌ترین مسیر از استیت شروع به استیت‌های پایانی، در ساختار گراف MDP مربوطه است. توجه داشته باشید که از استیت شروع به هر ترمینال، یک «کوتاه‌ترین مسیر» وجود دارد و در میان این کوتاه‌ترین‌ها، آن مسیری که کمترین طول ($\mathcal{L}$) را دارد مورد نظر ما است.

- فرض کنید دو محیط شبکه‌ای (grid) با ابعاد ۱۰ × ۱۰ و ۱۰۰ × ۱۰۰ داریم. در هر کدام از این دو محیط ۴ خانه با پاداش +۱ وجود دارد و تمامی سایر خانه‌ها دارای پاداش صفر هستند. همچنین فرض کنید که توزیع‌های احتمالی دلخواه $T_1(s'|s,a)$ و $T_2(s'|s,a)$ به ازای تک‌تک جفت استیت‌ـ اکشن‌های ممکن بر این محیط حاکم هستند و ما از این توزیع‌ها مطلع هستیم. حال اگر الگوریتم Value Iteration را روی این محیط‌ها با ضریب تخفیف‌های برابر $\gamma = 0.9$ اجرا کنیم و شرط توقف را این قرار دهیم که تغییرات $V(s)$ در دو مرحله متوالی کمتر از 0.001 باشد، الگوریتم در محیط بزرگ‌تر تعداد دفعات (Iteration) بیشتری اجرا می‌شود و دیرتر متوقف خواهد شد.
- در یک MDP که به صورت $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R, s_0)$ تعریف شده است، می‌دانیم که تابع R آن به صورت $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ است. فرض کنید که یک MDP جدید به صورت $\mathcal{M}' = (\mathcal{S}, \mathcal{A}, T, R', s_0)$ تعریف می‌کنیم. تابع پاداش جدید را از تبدیل affine تابع R می‌سازیم، به این معنا که: $R' = aR + b$ حال فرض کنید که سیاست‌های بهینه‌ی $\mathcal{M}$ و $\mathcal{M}'$ را به ترتیب π_* و π'_* بنامیم. ادعا می‌کنیم که $\pi_* = \pi'_*$.
- در مسائل RL دو دسته‌ی کلی وجود دارد: episodic و sequential. تفاوت اصلی این دو دسته مسئله در این است که در مسائل episodic دنباله‌ی استیت‌ها و اکشن‌های اتخاذشده، به چند قسمت (episode) تقسیم می‌شود و در حقیقت هر کدام از این قسمت‌ها پس از تعدادی گام محدود به پایان می‌رسد. این در حالی است که در دسته مسائل sequential کل سلسله‌ی تعاملات ایجنت و محیط به صورت یک‌جا در نظر گرفته می‌شود و تقسیم‌بندی‌ای در آن وجود ندارد. همچنین مفهومی وجود دارد به اسم «عایدی موردانتظار» یا expected return که در اصل تابعی است از پاداش‌های بدست‌آمده پس از یک زمان و آن را با G_t نشان می‌دهند. یک تعریف ساده برای این تابع در مسائل episodic به صورت زیر است:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T. \quad (1)$$

که در آن T لحظه‌ی پایان اپیزود است. همچنین برای دسته مسائل sequential از آنجایی که تعداد پاداش‌ها نامتناهی است، معمولا این فرمول را به صورت «تخفیف‌دار» یا discounted در نظر می‌گیرند:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots. \quad (2)$$

در فرمول (۲) لازم است که $0 \leq \gamma \leq 1$. ادعا می‌کنیم که می‌توان بر هر دوی این دسته مسائل از یک notation یکسان برای نوشتن تابع G_t استفاده کرد (راهنمایی: در صورتی که این گزاره از نظر شما درست است، باید سعی کنید نشان دهید که در اصل می‌توان یکی از دسته‌ها را به فرم دسته دیگر نوشت).
- سیاست قطعی π را در نظر بگیرید. اثبات کنید اگر سیاست جدید π' به صورت حریصانه از V^π بدست آمده باشد آنگاه π' بهتر یا مساوی π است، یا به عبارتی برای تمام حالت‌ها داریم $V^{\pi'}(s) \geq V^\pi(s)$.
همچنین اثبات کنید اگر تساوی برای تمام حالت‌ها برقرار باشد، آنگاه π' حتما سیاست بهینه است.

۳. (۳۰ نمره) بتمن که توانسته بود با کمک شما، سیاست مناسبی را برای حل معماهای ریدلر در پیش بگیرد، بمب او را خنثی و خودش را نیز دستگیر کرد.

پس از ختم به‌خیر شدن این ماجرا، بتمن تصمیم گرفته تا قابلیتی جدید به ماشین معروفش، بت‌موبیل (Batmobile)، اضافه کند. او می‌خواهد تا ماشینش را طوری تمرین دهد که بتواند در شهر بگردد و مکان‌هایی که در آن‌ها بمبی کار گذاشته‌شده را شناسایی کند.
او برای تمرین دادن بت‌موبیل، یک محیط شبیه‌سازی در مقرّش، Batcave، ساخته است که در این محیط در برخی نقاط بمب‌هایی واقعی و در برخی دیگر بمبی قلابی وجود دارند. اگر برایتان سوال است، به لطف آموزش‌های قبلی، بت‌موبیل این قابلیت را دارد که بمب‌هایی را که در فاصله‌ای کمی از اطرافش قرار دارند را به صورت خودکار تشخیص دهد. بنابرین با صرف قرارگیری در خانه‌ی مربوط به یک بمب، تشخیص آن خودبه‌خود انجام خواهد شد. البته این سیستم هوشمندی زیادی ندارد و نمی‌تواند بمب‌های قلابی را از واقعی تشخیص دهد.
در نتیجه، هر گاه بت‌موبیل به محل یکی از این بمب‌های قلابی برسد، ۱۰۰ امتیاز منفی دریافت می‌کند و هرگاه به مکان یک بمب واقعی برسد، ۱۰۰ امتیاز مثبت دریافت می‌کند. او همچنین با استفاده از داده‌هایی که از مکان‌های بمب‌گذاری‌های قبلی ریدلر و جوکر، Joker، جمع‌آوری کرده و با در نظر گرفتن موقعیت مکانی نقاط پرتردد شهر، این محیط را به صورت یک شبکه مربعی (grid) ۵ × ۵ طراحی کرده است. در ابتدا بت‌موبیل در وسط این شبکه قرار دارد و موقعیت مکانی آن (۰, ۰) است.
بتمن تصمیم دارد تا به کمک روش feature-based Q-learning این آموزش را انجام دهد. او برای این کار، استیت‌های داخل محیط را با موقعیت مکانی بت‌موبیل مدل کرده است. به عنوان مثال استیت شروع بت‌موبیل، به صورت بردار $s_0 = (x_0, y_0) = (0, 0)$ در نظر گرفته شده است. او همچنین یک فیچر دیگر را در نظر گرفته که صرفا به هر اکشن یک عدد را نسبت می‌دهد. به عبارت ساده‌تر، از آنجایی که بت‌موبیل در این محیط می‌تواند به راست، بالا، چپ و پایین حرکت کند، او اولاً این حرکات را به ترتیب با، R، U، L و D نشان می‌دهد و در ثانی، برای این حرکت‌ها به ترتیب مقادیر ارزش ۱، ۲، ۳ و ۴ را در نظر گرفته است. بنابرین، فرضاً اگر اکشن «حرکت به چپ را انجام دهد» ارزش ۳ به آن تعلق می‌گیرد و این مقدار را با $f_a(L) = 3$ نشان می‌دهیم.
حال، اگر موقعیت بت‌موبیل را در استیت (خانه) s با $s = (x_s, y_s)$ نشان دهیم، تابع مقدار استیت‌ـاکشن آن را به صورت زیر می‌توانیم تعریف کنیم:

$$Q(s, a) = w_x x_s + w_y y_s + w_a f_a(a)$$

. که در آن، طبق توضیحات قبلی $f_a(R) = 1, f_a(U) = 2, f_a(L) = 3, f_a(D) = 4$.

	s	a	s'	Reward
Episode 1	(۰, ۰)	R	(۱, ۰)	0
	(۱, ۰)	U	(۱, ۱)	0
	(۱, ۱)	U	(۱, ۲)	-100
Episode 2	(۰, ۰)	R	(۱, ۰)	0
	(۱, ۰)	R	(۲, ۰)	0
	(۲, ۰)	D	(۲, −۱)	+100
Episode 3	(۰, ۰)	U	(۰, ۱)	0
	(۰, ۱)	L	(−۱, ۱)	+100
	-	-	-	-

جدول ۱: تجربه‌های بدست آمده از محیط

پس از اینکه بتمن، بت‌موبیل را چند اپیزود در محیط آموزش می‌دهد، داده‌های جدول ۱ را جمع‌آوری می‌کند. او که از توانایی شما در انجام محاسبات سوال قبلی خوشش آمده است، این بار از شما می‌خواهد که با

اجرای الگوریتم Approximate Q-learning مقادیر نهایی وزن‌های فیچرهایی که برای مسئله در نظر گرفته بود را محاسبه کنیدر(مقادیر اولیه وزن‌ها را صفر در نظر بگیرید). در محاسبات خود، صرفا $Q(s,a)$هایی که تغییر می‌کنند را بنویسید. این مقادیر، یعنی $w = (w_x, w_y, w_a)$، چه معنایی دارند؟ توضیح دهید. $(\gamma = 1, \alpha = 0.5)$.

۴. (۲۰ نمره) در تعطیلات تابستانی علی به سفر رفته است. در طول مدتی که در سفر است در یکی از سه اتاق حمام، پذیرایی و اتاق خواب قرار دارد. در هر حرکت به احتمالات موجود در جدول به اتاق‌های مجاور میرود و با حرکت از هر اتاق به اتاق دیگر از درجه تمیزی علی یکی کم میشود. از آنجایی که علی به تمیزی خود بسیار اهمیت میدهد به محض اینکه تمیزی او به یک برسد در حرکت بعدی به حمام میرود و تمیزی او به ۵ میرسد.

Start	Action	End	Probability	reward
bathroom	to living room	living room	0.8	3
bathroom	to living room	bedroom	0.2	-1
bathroom	to bedroom	living room	0.3	1
bathroom	to bedroom	bedroom	0.7	4
bedroom	to living room	living room	0.9	3
bedroom	to living room	bathroom	0.1	-2
bedroom	to bathroom	bathroom	1	2
living room	to bedroom	bedroom	0.5	3
living room	to bedroom	bathroom	0.5	-1
living room	to bathroom	bathroom	1	2

با استفاده از ضریب تخفیف های 0.8 و 0.2 به سوالات زیر پاسخ دهید.

(آ) فضای حالت‌ها را تعریف کنید. رابطه به روزرسانی معادله بهینه بلمن را برای تابع ارزش حالتها بنویسید.

(ب) میخواهیم از روش Q-value iteration برای بروزرسانی Q-value ها استفاده کنیم. ابتدا رابطه‌ی بروزرسانی را برای Q-value ها بنویسید. سپس با درنظر گرفتن مقدار صفر برای مقدار اولیه‌ی Q-value ها پالیسی بهینه برای هر کدام از استیت‌ها را به دست آورید.

(ج) با توجه به قسمت قبل در کدام حالت ضریب تخفیف، رفتار علی به رفتار حریصانه نزدیک‌تر است؟ توضیح دهید.

۵. (۲۵ نمره) در بسیاری از مسائل، مانند سیستم درمانی ما نمی‌توانیم تمام پالیسی‌های ممکن را انجام دهیم و باید از دیتاهای یک پالیسی استفاده کنیم و از آن برای محاسبه value سایر پالیسی‌ها استفاده کنیم. برای این منظور باید با محاسبه اختلاف value پالیسی‌های مختلف در یک fixed horizon MDP آشنا شویم. یک fixed horizon MDP، MDPای است که بعد از H مرحله، ریست میشود، که به H ، horizon آن MDP گفته می‌شود. در این حالت هیچ ضریب تخفیفی وجود ندارد ($\gamma = ۱$) و پالیسی‌ها می‌توانند non stationary باشند یعنی اکشن‌های یک پالیسی علاوه بر استیت به تایم استپ نیز مرتبط شوند. اگر $x_t \sim \pi$ توزیع استیت‌ها در تایم استپ t ($۱ \leq t \leq H$) روی پالیسی π باشند و $V_t^\pi(x_t)$ و $Q_t^\pi(x_t)$ به ترتیب value function و Q value نسبت داده شده به پالیسی π در تایم استپ t باشند، عبارت زیر را اثبات کنید. ($\mathbb{E}_{x_t \sim \pi} V_t^\pi(x_t)$ به معنی میانگین ولیو ها در تایم استپ t است در حالی که x_t از پالیسی π پیروی میکند و $\pi_۱$ و $\pi_۲$ دو پالیسی متفاوت هستند.)

$$\mathbb{E}_{x_1 \sim \pi_2}(V_1^{\pi_1}(x_1) - V_1^{\pi_2}(x_1)) = \sum_{t=1}^{H} \mathbb{E}_{x_t \sim \pi_2}(Q_t^{\pi_1}(x_t, \pi_1(x_t, t)) - Q_t^{\pi_1}(x_t, \pi_2(x_t, t)))$$

سوالات عملی (۱۵۰ نمره)

۱. (۹۰ نمره) برای پاسخ به این سوال به پوشه سوالات عملی بخش Q1 مراجعه کنید.

۲. (۶۰ نمره)

در این تمرین قصد پیاده‌کردن الگوریتم Q-learning و Approximate Q-learning برای بازی pacman را داریم.

برای اجرای کد بخش گرافیک و تست این برنامه نیاز به پایتون نسخه ۲.۷.۱۶ است که می‌توانید آن را برای Windows/MacOS از اینجا دانلود و نصب کنید و برای لینوکس از این لینک استفاده کنید. (این مورد برای اجرا شدن گرفیک است و لازم نیست که موارد جدیدی را syntax این ورژن یاد بگیرید چون احتیاجی به آن در اینجا نخواهد بود.)

برای بازی کردن با نقشه‌های مختف می‌توانید از دستورات زیر استفاده کنید:

```
python pacman.py -l smallGrid
python pacman.py -l mediumGrid
python pacman.py -l mediumClassic
```

در بین فایل هایی که در اختیارتان قرار دارند، تنها نیاز است که فایل qlearningAgents.py را ویرایش کنید و متود های ناکامل آن را تکمیل کنید. برای انجام این تمرین، مطالعه کد در این دو فایل کافی است و نیازی به مطالعه کد در فایل های دیگر را ندارید. دو فایلی که نیاز به خواندن آنها است و به تغییر دادن نیاز ندارند:

- learningAgents.py: کلاس پایه QLearningAgent را که عامل (agent) شما از آن ارث‌بری می‌کند را تعریف می‌کند.
- featureExtractors.py: کلاس‌های برای استخراج feature برای هر جفت (حالت، عمل) را تعریف خواهد کرد.

util.Counter در واقع یک دیکشنری است که برای کلیدی که در دیکشنری وجود نداشته باشد مقدار صفر را برمی‌گرداند و از آن برای سادگی در پیاده‌سازی می‌توانید استفاده کنید. این داده ساختار در فایل util.py پیاده شده است و نیازی به مطالعه کد آن در این فایل ندارید.

(آ) بخش اول: در ابتدا یک عامل یادگیرنده با الگوریتم Q-learning را باید ایجاد کنیم. برای این‌کار، کلاس QlearningAgent را تکمیل کنید.

(ب) بخش دوم: با تکمیل کردن کلاس قبل، با استفاده از دستور زیر عامل pacman خود را برای بازی با نقشه کوچک تست کنید.

```
python pacman.py -p PacmanQAgent -x 2000 -n 2010 -l smallGrid
```

در این دستور عامل Q-learning برای ۲۰۰۰ بار train خواهد شد و برای ۱۰ بازی بعدی تست خواهد شد. (n- تعداد کل دفعات بازی و x- تعداد دفعات بازی برای training را نشان می‌دهد.) دقت کنید که در مرحله تست مقدار ϵ و نرخ یادگیری برابر صفر خواهد شد تا دیگر عامل قابلیت یادگیری و exploration را نداشته باشد. در این‌جا نیز دقت کنید که در این دستور از عامل در یادگیری از کلاس PacmanQAgent است که برای آن پارامترهای مناسب (alpha،gamma،epsilon) برای یادگیری بازی تعیین شده است. هرچند شما نیز میتوانید با استفاده از دستور زیر یادگیری را برای پارامترهای دیگر نیز بررسی کنید:

```
python pacman.py -p PacmanQAgent -x 2000 -n 2010 -l smallGrid \
-a alpha=0.7, gamma=0.3, epsilon=0.1
```

در دستور بالا، pacman باید در حداقل ۸ بازی برنده بشود. دستور بخش قبل را برای mediumGrid و mediumClassic نیز اعمال کنید. آیا عملکرد آن در اینجا به خوبی بخش قبل خواهد بود؟ (علت آن نیز توضیح داده شود)

(ج) بخش سوم: در این قسمت قصد پیاده‌کردن عاملی را داریم که با استفاده از الگوریتم Approximate Q-learning بتواند بازی را انجام دهد و آن را در کلاس ApproximateQAgent پیاده کنید. با استفاده featureExtractor می‌توانید feature های مورد نیاز برای اجرا الگوریتم را بدست آورید. در اینجا دو نوع featureExtractor پیاده شده است:

i. IdentityExtractor : که به ازای هر جفت (حالت، اکشن) یک بردار تک عضوی feature می‌دهد.
ii. SimpleExtractor : که به ازای هر جفت (حالت، اکشن) یک بردار feature با بیش از یک عضو را خروجی خواهد داد.

کلاس ApproximateQAgent را تکمیل کنید و بعد از آن، برای اطمینان از صحت پیاده‌سازی، اگر از IdentityExtractor استفاده کنید، باید همان عملکرد الگوریتم Q-learning را در حالت smallGrid داشته باشد:

```
python pacman.py -p ApproximateQAgent -x 2000 -n 2010 \
-a extractor=IdentityExtractor -l smallGrid
```

و بعد از آن می‌توانید با استفاده از SimpleExtractor نحوه عملکرد الگوریتم را در حالت های mediumGrid و mediumClassic بررسی کنید.

```
python pacman.py -p ApproximateQAgent -x 50 -n 60 \
-a extractor=SimpleExtractor -l smallGrid
```

```
python pacman.py -p ApproximateQAgent -x 50 -n 60 \
-a extractor=SimpleExtractor -l mediumGrid
```

در دستورات بالا، به‌ازای هرحالت زمین بازی pacman باید در حداقل ۸ بازی از ۱۰ بازی در مرحله تست برنده بشود.
آیا در حالت mediumClassic، عامل pacman در بعضی از مواقع رفتاری عجیب از خود نشان نمی‌دهد؟ در صورت وجود آن، علت آن را چه چیزی می‌دانید؟ (راهنمایی: به feature های استخراج شده از بازی دقت نمایید)

نحوه ارسال جواب: فایل ویرایش شده qlearningAgents.py را به‌همراه یک فایل متنی/pdf که در آن پاسخ به سوالات قرار دارد را در پوشه سوالات عملی بخش Q2 قرار دهید.

نمره‌دهی: علاوه بر معیارهای گفته شده در متن سوال، تعدادی تست نیز بر کد شما نیز اعمال خواهد شد.