به نام خدا

علوم اعصاب: یادگیری، حافظه، شناخت — شماره درس: ۲۵۶۱۴

تمرین عملی سوم

موعد تحویل: ۲۳ آذر — ترم پاییز ۱۳۹۷

Reinforcement Learning

هدف از این تمرین طراحی یک مدل decision making مبتنی بر reinforcement learning است. برای این تمرین یک کلاس به اسم Shadlen در اختیار شما قرار می گیرد و باید کلاسی به اسم Monkey طراحی کنید که در محیطی که Shadlen فراهم آورده، تصمیم‌گیری می کند و پاداش گرفته و یا جریمه می شود و این فرایند تا رسیدن به بهینه‌ترین تصمیم‌ها ادامه می یابد.

تسک طراحی شده بر اساس [۱] می باشد. در هر مرحله از تصمیم‌گیری کلاس Shadlen چهار شی از ده شی موجود را به میمون نمایش می دهد و میمون باید بر اساس مجموع ارزش این اشیا، بین ۰ و ۱ انتخاب کند. با تصمیمی که میمون در این محیط می گیرد وارد حالت بعدی شده و پاداشی هم می گیرد.

خواسته ها:

کلاسی به اسم Monkey طراحی کنید و یک شبکه LSTM را به عنوان شبکه‌ی پیش‌بینی Q در آن قرار دهید. ورودی این شبکه s و خروجی آن Q است. خروجی شبکه دوتایی است که نشان دهنده تصمیم چپ یا راست (۰ یا ۱) است.

ورودی LSTM، یک ماتریس 10×5 می باشد که حاوی چهار شی رندوم انتخاب شده و مرحله تصمیم‌گیری (عمق فعلی در درخت) است که به صورت بردار one-hot دهتایی کد شده اند.

با استفاده از الگوریتم معرفی شده، میمون را در شرایط reinforcement learning قرار دهید تا تصمیم‌گیری در این محیط را یاد بگیرد.

کلاس Shadlen به عنوان emulator عمل کرده و برای میمون state و پاداش‌ها و رسیدن به ترمینال را مشخص می کند.

عمل یادگیری را در تعداد اپیزود کافی انجام دهید و نمودار میزان پاداش میانگین (در پنجره های ۲۰۰ اپیزودی) را در طول یادگیری رسم نمایید.

میزان epsilon را به صورت نمایی و از یک مقدار اولیه معقول کاهش دهید. تاثیر استراتژی های متفاوت بر روی نتایج به چه شکل است؟ میزان gamma را تغییر دهید و نتیجه آن را مشاهده و تفسیر کنید. ساختمان LSTM را تغییر دهید و تاثیر آن بر روی نتایج را مشاهده کنید.

تابع های موردنیاز برای استفاده:

تابع reset() یک state را به عنوان خروجی می دهد.

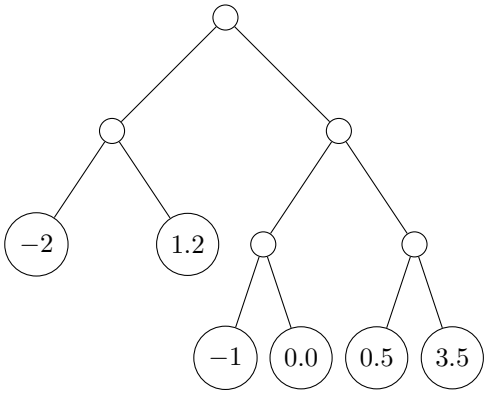
تابع response() که state ، پاداش و وضعیت رسیدن به ترمینال ها را مشخص می کند.

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

Algorithm 1 Deep Q-learning

for episode $= 1, M$ **do**
  Initialize sequence $s_0 = \{x_1\}$
  for $t = 1, T$ **do**
    Predict Q_t from the current state, s_t
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and s_{t+1}
    Predict Q_{t+1} from s_{t+1}
    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 1
    $s_t \leftarrow s_{t+1}$
    Break if the algorithm reaches the terminal state
  end for
end for

استراتژی در نظر گرفته شده در کلاس Shadlen به صورت زیر است. در ترمینال ها، پاداش ها و جرایم مشخص شده اند، در سایر راس ها میزان پاداش صفر است.



در هر راس، یال جدا شده ی سمت راست، تصمیم گیری درست و یال سمت چپ تصمیم غلط را نشان می دهد که به وضعیت های متفاوتی منتج می شوند.

References

[1] Yang, T., & Shadlen, M.N. (2007). Probabilistic reasoning by neurons. Nature, 447, 1075-1080.

[2] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M.A. (2013). Playing Atari with Deep Reinforcement Learning. CoRR, abs/1312.5602.