پروژهٔ نهایی هوش مصنوعی

پدرام طاهری، ۶۱۰۳۹۵۱۲۳

Abstract:

یادگیری با استفاده از Q-Learning و سیاست epsilon-greedy برای انتخاب حرکت در زمان یادگیری انجام شده است. تابع پاداش، دست‌نویس و بهینه شده است. از Learning Rate Decay نیز برای بهینه کردن فرآیند یادگیری استفاده شده. چندین بار یادگیری انجام می‌شود و از سیاست یادگرفته‌شده، ارزیابی به عمل می‌آید. در نهایت بهترین سیاست انتخاب می‌شود. الگوریتم در نهایت به درصد موفقیت نزدیک به ۵۰ روی زمین 8x8 می‌رسد.

Initialization:

برای مقادیر شروع در جدول Q، از مقداری رندوم در بازهٔ 0.49, 0.51 استفاده شد. بازه باید کوچک باشد تا بین حرکت‌های مختلف، در آغاز، تفاوت معنی‌داری وجود نداشته باشد و نیز متفاوت و تصادفی‌است که الگوریتم بتواند با اجراهای متعدد حالات مختلفی را بررسی کند و به بهینه‌ترین حالتِ ممکن در این زمینِ تصادفی نزدیک شود.

مقادیر ثابت استفاده شده نیز به صورت زیر هستند:

```
LR_DECAY = 0.1
LR_SCHED = 1000
ITERATIONS = 10000
GAMMA = 0.9
LEARNING_RATE = 0.1
```

هر LR_SCHED(۱۰۰۰) بازی، نرخِ یادگیری در LR_DECAY(۰.۱) ضرب می‌شود.

Q-Value:

مقادیر خانه‌های سوراخ و هدف در این جدول، از ابتدا با همان مقدار رندوم پر شده و تا انتها تغییری نمی‌کنند (زیرا هرگاه بازی به آن مراحل برسد متوقف می‌شود و تصحیح جدول Q خاتمه می‌یابد.)

Reward Function:

در تابع پاداش، برای یادگیری سریع‌تر، نزدیکی به هدف گنجانده شد، به این صورت که هرگاه حرکتی موجب نزدیکی ما به هدف شود، یک پاداش می‌گیریم. هرگاه حرکتی موجب افتادن ما در سوراخ شود، -۱۰ پاداش و هرگاه به هدف برسیم، ۵۰ پاداش می‌گیریم. مقادیر سوراخ و هدف باید به طرز قابل توجهی بالاتر از مقادیر نزدیک شدن به هدف باشند تا الگوریتم از سوراخ‌ها دوری کند و به هدف متمایل شود. اگر به دور شدن از هدف امتیاز منفی داده می‌شد، می‌توانست در یافتن بهینه‌ترین مسیر کمک‌کننده باشد اما ممکن بود ما را به سمت چاله بکشاند. این تابع، تضمین‌کنندهٔ دوری ما از چاله‌ها حتی در صورت نزدیک شدنِ اتفاقی (در صورت لیز خوردن) به آن‌هاست.

Q Evaluation:

تابع Q یادگرفته شده به طرز عجیبی گاها مسیر مستقیم را انتخاب نمی‌کند. به نظر می‌رسد (اکثر مواقع) مسیرِ بهینه‌ای که یاد گرفته می شود، رفتن تا انتها به سمت راست و سپس تا انتها پایین رفتن است. اما پایین رفتن پس از رسیدن به گوشهٔ نقشه به گونهٔ عجیبی اتفاق می‌افتد. با دادنِ (گاها) دستور رفتن به سمت راست(به سمت دیوار)، به جای پایین. حدس من این است که بازی طوری کار می‌کند که رفتن به سَمت راست، خطر افتادن به چاله و لیز خوردن را کمتر می‌کند. نیز به این دلیل که الگوریتم برای کمترین تعداد حرکات بهینه نشده، ابایی از اتلاف وقت در خانه‌ها و فرستادن دستورهای زیاد ندارد. برای همین به جای رفتن به پایین، سمت راست را نشانه می‌رود. گاها رفتارهای عجیبی از تابع Q برای رفتن به سمت سوراخ (با درصد بالای قاطعیت) نیز مشاهده می‌شود که به گمانم به دلیل کم بودن تعداد اجراها و تصادفی بودن محیط به وجود آمده‌اند. (برای مثال در خانهٔ سمت چپ هدف که بالایش دارای سوراخ است، گاهی سیاست یادگرفته شده دستور به بالا رفتن می‌دهد. گمان می‌کنم پیش آمده است که دستورِ بالا رفتن، منجر به افتادن در هدف شده باشد.

لینک گیت‌هاب تمارین و پروژه:

https://github.com/pdrmtaheri/ArtificialIntelligenceCourse