Relatório: Solução do Problema CartPole utilizando Q-learning

1. Introdução

O problema do CartPole consiste em equilibrar um pêndulo invertido em cima de um carrinho, que pode se mover para a esquerda ou para a direita. O objetivo é manter o pêndulo em pé pelo maior tempo possível, maximizando a recompensa acumulada. Para resolver este problema, foi implementado um agente de aprendizado por reforço utilizando o algoritmo Q-learning, uma técnica clássica que permite ao agente aprender uma política ótima por meio de experiências e das equações de otimalidade de Bellman.

2. Modelagem do Problema como MDP

O problema do CartPole pode ser modelado como um Processo de Decisão de Markov (MDP), que é definido pelos seguintes componentes:

- Estados (S): O estado é representado por quatro variáveis contínuas - posição do carrinho, velocidade do carrinho, ângulo do pêndulo e velocidade angular do pêndulo.
- Ações (A): O agente pode aplicar uma força para a esquerda ou para a direita, ou seja, tem duas ações disponíveis.
- Recompensas (R): A cada passo de tempo, o agente recebe uma recompensa de +1 se o pêndulo permanece em pé.
- Transições (T): As transições entre os estados são determinadas pelas leis físicas do ambiente, dependendo da ação tomada e do estado atual.

Objetivo: Aprender uma política que maximize a soma das recompensas ao longo do tempo, mantendo o pêndulo equilibrado pelo maior tempo possível.

3. Discretização dos Estados

Como o problema do CartPole possui estados contínuos, foi necessário discretizar os estados para aplicar o algoritmo Q-learning, que trabalha com tabelas de estado-ação. Para isso, cada variável de estado foi dividida em 24 faixas (bins), transformando o espaço de estados contínuo em um espaço de estados discretos que pode ser gerenciado por uma tabela Q.

4. Algoritmo de Q-learning

O Q-learning é um algoritmo de aprendizado por reforço off-policy que busca aprender uma função de valor Q(s, a), que representa a expectativa de recompensa futura para cada par estado-ação. A cada iteração, a tabela Q é atualizada usando a seguinte

equação de Bellman:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

Onde:

- Q(s, a) é o valor atual da função Q para o par estado-ação (s, a).
- α é a taxa de aprendizado, que controla o quanto do novo valor será incorporado ao valor Q existente (0 < α ≤ 1).
- r é a recompensa imediata recebida ao executar a ação a no estado s.
- γ é o fator de desconto, que define a importância das recompensas futuras em relação à recompensa imediata (0 ≤ γ ≤ 1).
- s' é o próximo estado alcançado após executar a ação a.
- max_a' Q(s', a') é o valor máximo da função Q para o próximo estado s', considerando todas as possíveis ações a' a partir de s'.

5. Aplicação da Equação de Bellman no Q-learning

Durante o treinamento do agente, a Equação de Bellman foi usada para atualizar iterativamente a tabela Q, de modo que o agente pudesse melhorar sua política. A cada passo de tempo, após tomar uma ação e observar o novo estado e a recompensa, o valor Q(s, a) era ajustado, aproximando-se cada vez mais da política ótima. Essa atualização permitiu ao agente aprender qual ação deveria ser escolhida em cada estado para maximizar a recompensa total acumulada ao longo do tempo.

A equação de Bellman, assim, garante que o agente esteja sempre "aprendendo" a partir de suas interações, ajustando os valores de Q para refletir a melhor estimativa da recompensa esperada. O objetivo é que a tabela Q converja para os valores verdadeiros da função de valor, o que corresponde ao aprendizado da política ótima para o problema do CartPole.

6. Política-gulosa

Para balancear entre exploração (tentar novas ações) e exploração (escolher a melhor ação conhecida), foi utilizada a política ε-gulosa. Inicialmente, a probabilidade de explorar (ε) é alta, mas é reduzida gradualmente ao longo do treinamento (ε decai a cada episódio), permitindo que o agente explore mais no começo e se concentre em aproveitar o que aprendeu posteriormente.

7. Treinamento do Agente

O treinamento do agente foi realizado ao longo de 10.000 episódios, onde em cada episódio o agente interage com o ambiente, coleta recompensas e atualiza a tabela Q com base nas experiências vividas. Durante cada episódio, o agente executa ações e observa o próximo estado e a recompensa recebida, aplicando a equação de Bellman

para melhorar a sua política.

A recompensa total acumulada em cada episódio foi registrada e, ao final do treinamento, um gráfico foi gerado para visualizar o progresso do agente. Conforme o agente aprende, espera-se que a recompensa total aumente, indicando que ele está conseguindo manter o pêndulo em pé por mais tempo.

8. Desempenho do Agente Treinado

Após o treinamento, o agente foi capaz de manter o pêndulo equilibrado por vários passos, evidenciando a eficácia do aprendizado por reforço com Q-learning. O comportamento do agente treinado foi demonstrado graficamente e o ambiente foi renderizado para visualizar o desempenho.

9. Possíveis Melhorias

Embora o Q-learning tenha se mostrado eficaz para resolver o problema do CartPole, uma possível melhoria seria a substituição da tabela Q por uma rede neural, como no Deep Q-Network (DQN). Isso permitiria lidar diretamente com estados contínuos, eliminando a necessidade de discretização e proporcionando maior escalabilidade para problemas mais complexos.

Outra melhoria possível é o ajuste dos hiperparâmetros, como a taxa de aprendizado e o fator de desconto, para otimizar ainda mais o desempenho do agente. Além disso, técnicas como Replay Buffer e Target Networks poderiam ser usadas para estabilizar e acelerar o aprendizado.

10. Conclusão

Neste trabalho, foi implementada uma solução para o problema do CartPole utilizando o algoritmo Q-learning, que permitiu ao agente aprender a manter o pêndulo invertido equilibrado ao aplicar forças à esquerda ou à direita. A solução focou no uso das equações de Bellman para atualizar os valores da tabela Q e encontrar a política ótima, maximizando a recompensa acumulada ao longo do tempo.

Embora a abordagem tenha sido bem-sucedida, existem diversas melhorias que poderiam ser exploradas, especialmente no uso de redes neurais para lidar com espaços de estados contínuos e mais complexos, mostrando o potencial do aprendizado por reforço profundo para problemas mais sofisticados.