IIA - Q Learning

Felipe Cadar Chamone

Novembro 2020

1 Introdução

Esse trabalho se trata da implementação e análise de sensibilidade de parâmetros do algoritmo Q-Learning. O Q-Learning é um algoritmo livre de modelo que foca em determinar a melhor ação a se tomar em cada estado, isso significa que para seu funcionamento precisamos de um espaço discreto de ações e estados, apesar de que pode ser facilmente adaptando dividindo espaços contínuos em intervalos. No nosso caso tentamos resolver um problema de planejamento de caminho de uma plataforma robótica em mapas conhecidos, então ambos os espaços de ações e estados são finitos e conhecidos.

2 Implementação

Para o funcionamento do algoritmo precisamos modelar os estados do problema na tabela de de Qvalues. Nesse trabalho foi modelado que o estado é constituído de 3 valores: posição X, posição Y e passos desde o último ponto de localização. Então nossa qTable vai ser do tamanho $L*C*W*A$, onde L e C são a quantidade de linhas e colunas no mapa, W é o máximo de passos permitido entre estações de localização e A é a quantidade de ações.

Com o modelo da tabala definido, é bem simples seguir a regra de atualização da equação do Q Learning:

$$Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha * (r_t + \gamma * \max_a Q(s_{t+1}, a) - Q(s_t, a_t))$$

Também definimos um nível de decaimento para o parâmetro de exploração, tal que a cada certo número de épocas ele é multiplicado por 0.999, desse modo usamos cada vez mais o aprendizado do agente.

O ciclo de treinamento possui duas etapas: teste e treino. Em toda época treinamos o agente de forma tradicional, usado o fator de exploração. A cada 100 épocas rodamos um teste, onde o agente não é atualizado e apenas usa seus conhecimentos para navegar no mapa. Assim podemos ter noção do desempenho do treinamento sem o ruído das ações aleatórias da fase de exploração.

3 Experimentos

O algoritmo possui 3 parâmetros: *Learning Rate*, *Discount Factor* e *Exploration Factor*. Para otimiza-los seguimos um protocolo padrão de análise de sensibilidade, mantendo todos os parâmetros fixos e alterando um de cada vez. Ao final escolhemos os melhores valores de cada parâmetro e rodamos os experimentos finais. Cada variação é executada 10 vezes e tomamos a média dos valores de recompensa acumulada em cada época para comparar os parâmetros.

O processo de achar os melhores parâmetros foi automatizado. Depois de rodar todos os experimentos, avaliamos as curvas de recompensa acumulada e escolhemos o parâmetro com a maior área sobre a curva.

Os parâmetros padrões usados são: *Learning Rate*: 0.1, *Discount Factor*: 0.7, *Exploration Factor*: 0.5 e Épocas: 10000.

Essa sessão apenas expõe os resultados gráficos, vamos analisa-los na sessão de Resultados. A seguir estão os resultados para dos 3 parâmetros de todos os mapas.

3.1 input0

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.3, *Exploration Factor*: 0.1 e *Discount Factor*: 0.1

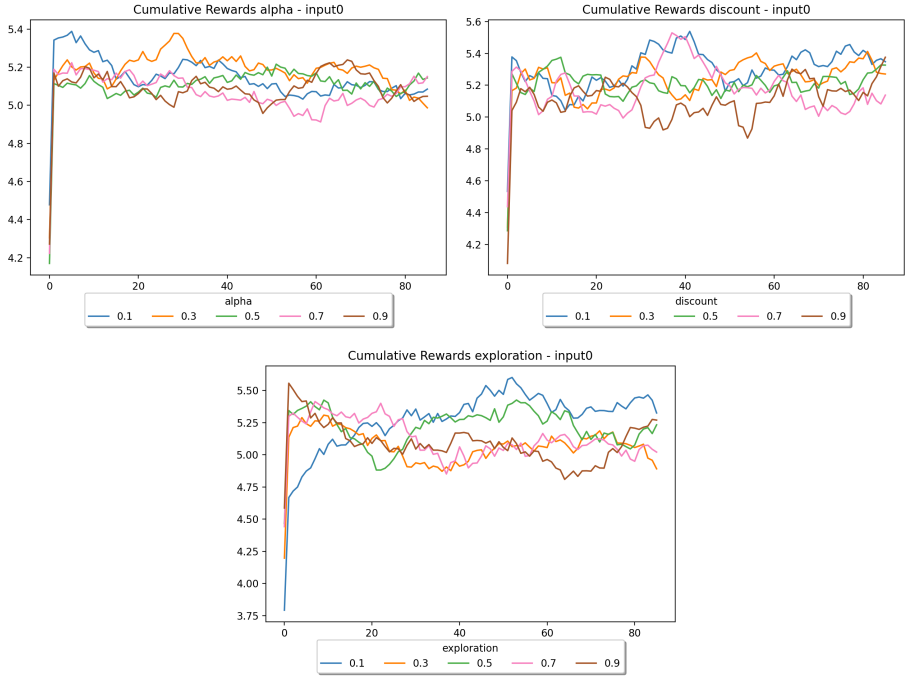

Figure 1: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.2 input1

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.9, *Exploration Factor*: 0.5 e *Discount Factor*: 0.7

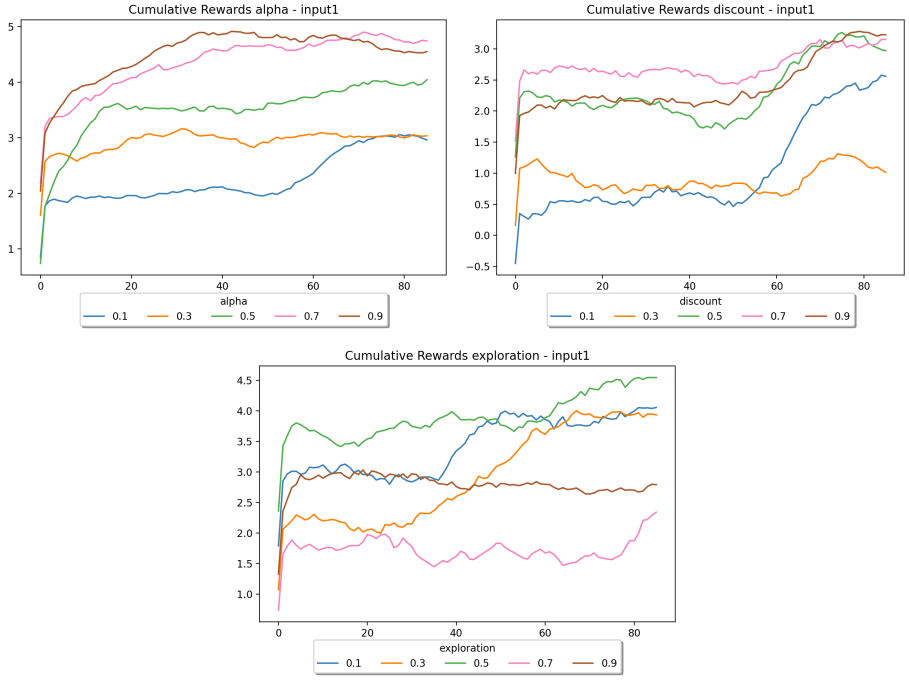

Figure 2: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.3 input2

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.1, *Exploration Factor*: 0.1 e *Discount Factor*: 0.5

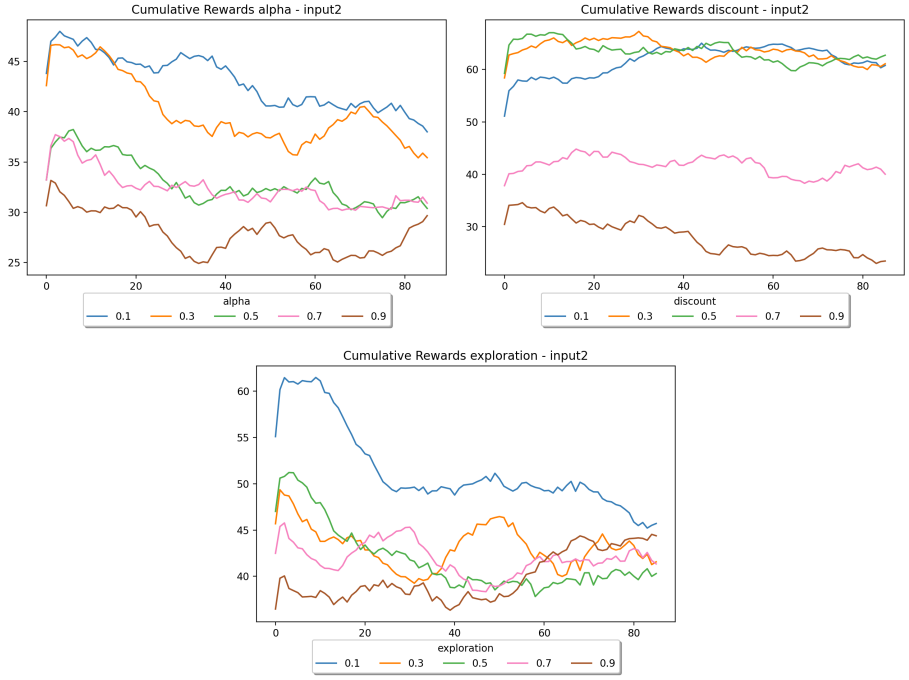

Figure 3: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.4 input3

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.3, *Exploration Factor*: 0.1 e *Discount Factor*: 0.5

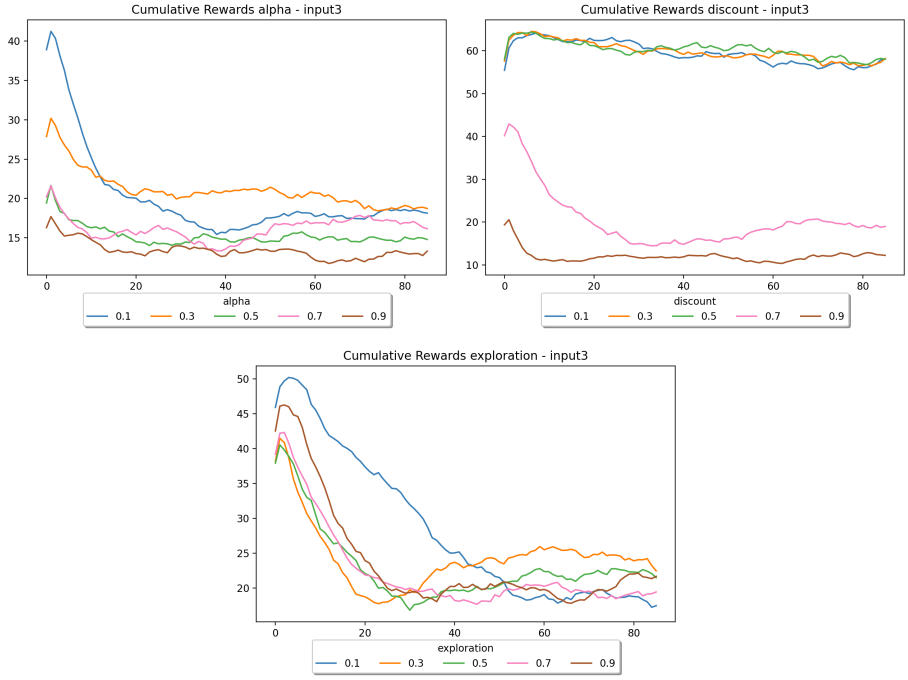

Figure 4: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.5 input4

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.7, *Exploration Factor*: 0.9 e *Discount Factor*: 0.9

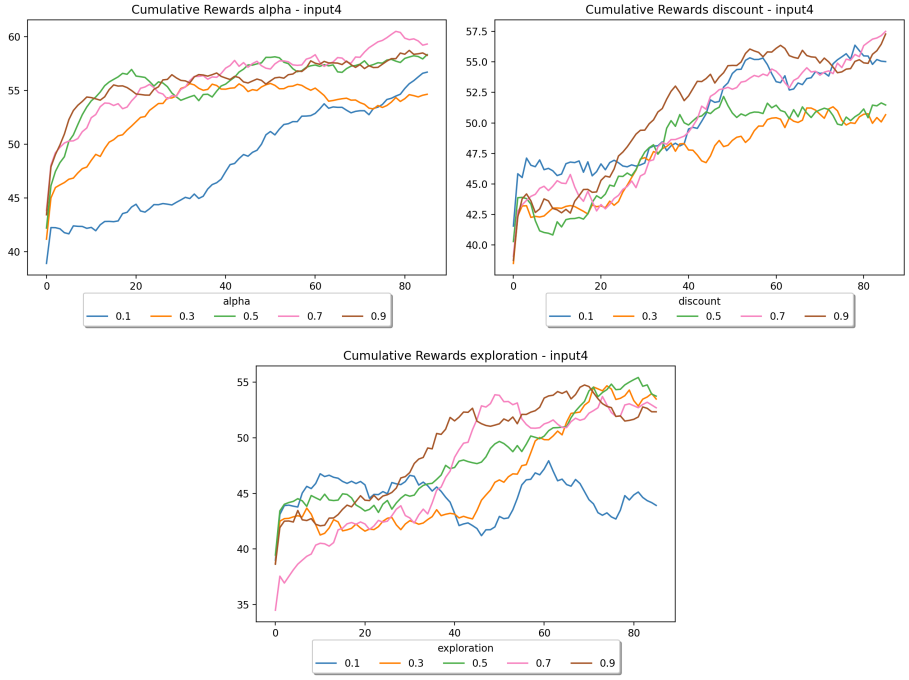

Figure 5: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.6 input5

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.1, *Exploration Factor*: 0.9 e *Discount Factor*: 0.5

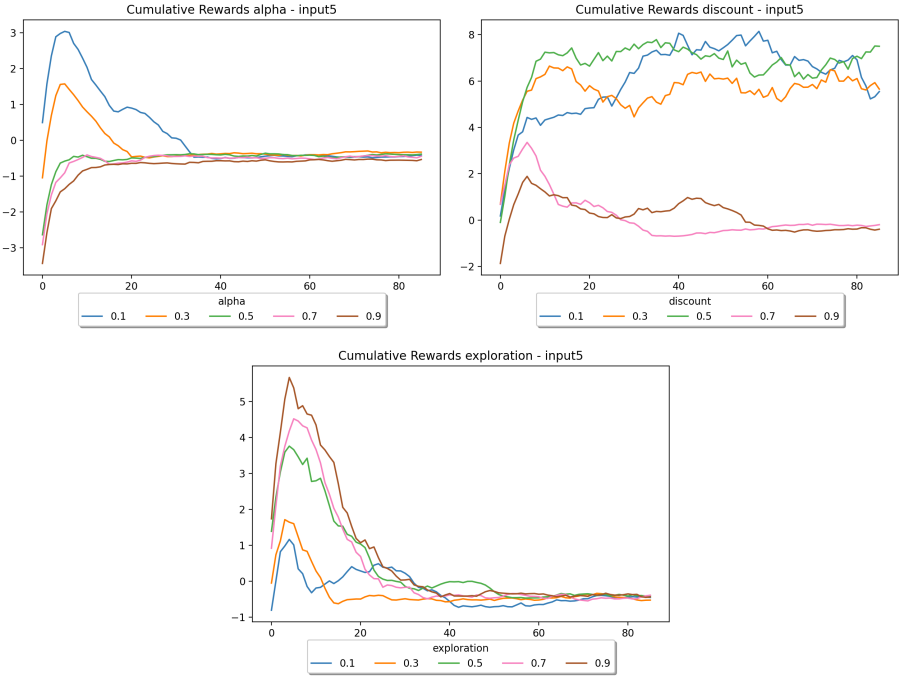

Figure 6: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.7 input6

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.7, *Exploration Factor*: 0.7 e *Discount Factor*: 0.5

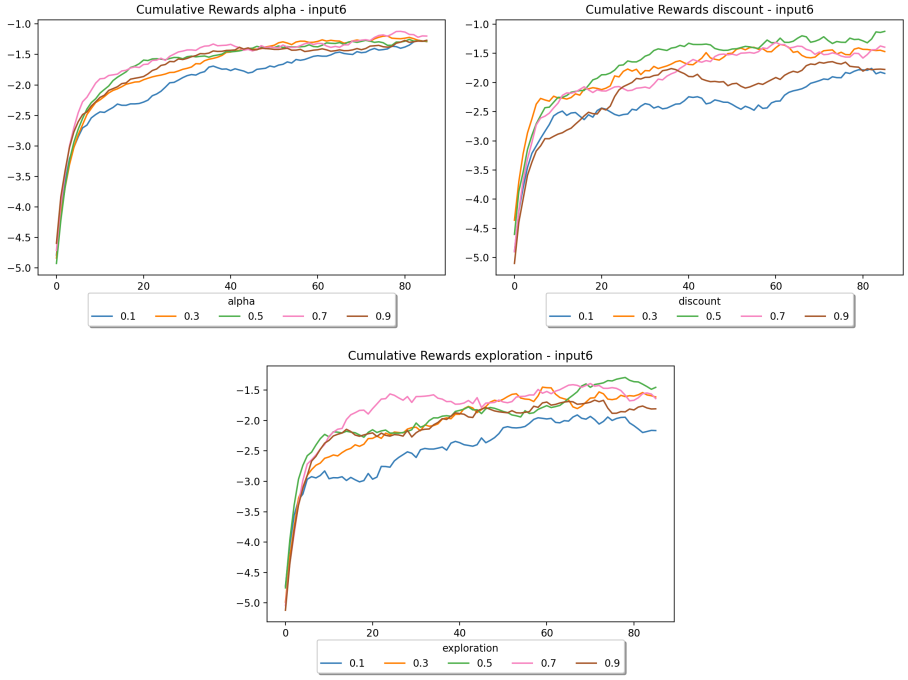

Figure 7: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

3.8 input7

Os parâmetros escolhidos pela heurística foram: *Learning Rate*: 0.9, *Exploration Factor*: 0.9 e *Discount Factor*: 0.1

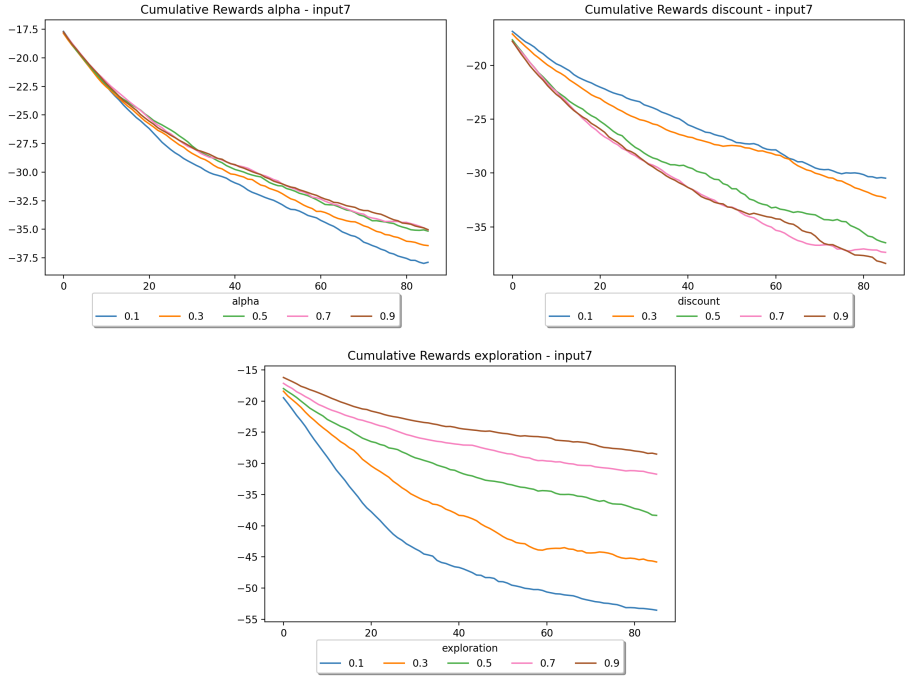

Figure 8: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*

Mapa	W	Learning Rate	Exploration Factor	Discount Factor
input0	3	0.3	0.1	0.1
input1	3	0.9	0.5	0.7
input2	3	0.1	0.1	0.5
input3	3	0.3	0.1	0.5
input4	2	0.7	0.9	0.9
input5	5	0.1	0.9	0.5
input6	3	0.7	0.7	0.5
input7	50	0.9	0.9	0.1

4 Resultados

Aqui temos um resumo dos parâmetros escolhidos

O primeiro mapa é o menor de todos, e por ser tão peque é bem possível de aprender bem com praticamente qualquer parâmetro, ainda mais que rodamos 10000 iterações de treino. Por esse motivo as curvas ficaram tão próximas, como podemos ver na figura 1.

Nos mapas 1 a 6, já começamos a ver um *Discount Factor* bem maior, mostrando que o tamanho do mapa, combinado ao baixo valor de W, precisa que o algoritmo "pensei no futuro", levando mais em consideração o valor de recompensa dos próximos passos exatamente por ter um número limitado de passos.

No mapa 7 podemos observar que o *Discount Factor* foi bem pequeno, isso se dá pelo alto valor de W, logo uma estratégia que pensa apenas na recompensa imediata pode funcionar bem.

Também podemos notar um padrão no Fator de exploração. Nos menor mapas, 0 a 3, ele é bem menor e nos mapas maiores, 4 a 7, praticamente todos atingem o máximo. Podemos dizer que com o aumento do tamanho do mapa precisamos passar mais tempo com um alto fator de exploração. Deves lembrar aqui que foi implementado um sistema de decaimento de exploração, então mesmo com valores iniciais tão altos conseguimos aproveitar o aprendizado do agente ao fim e passar para uma fase de *exploitation*.

Na figura 9 temos o desempenho final em cada mapa.

5 Conclusão

Com esse trabalho podemos perceber que, apesar da simplicidade do algoritmo ele consegui aprender a resolver a maioria dos mapas. Os mapas maiores, como o 6 e o 7, não se saíram bem pela quantidade de estados e, talvez, por uma falha de otimização de parâmetros.

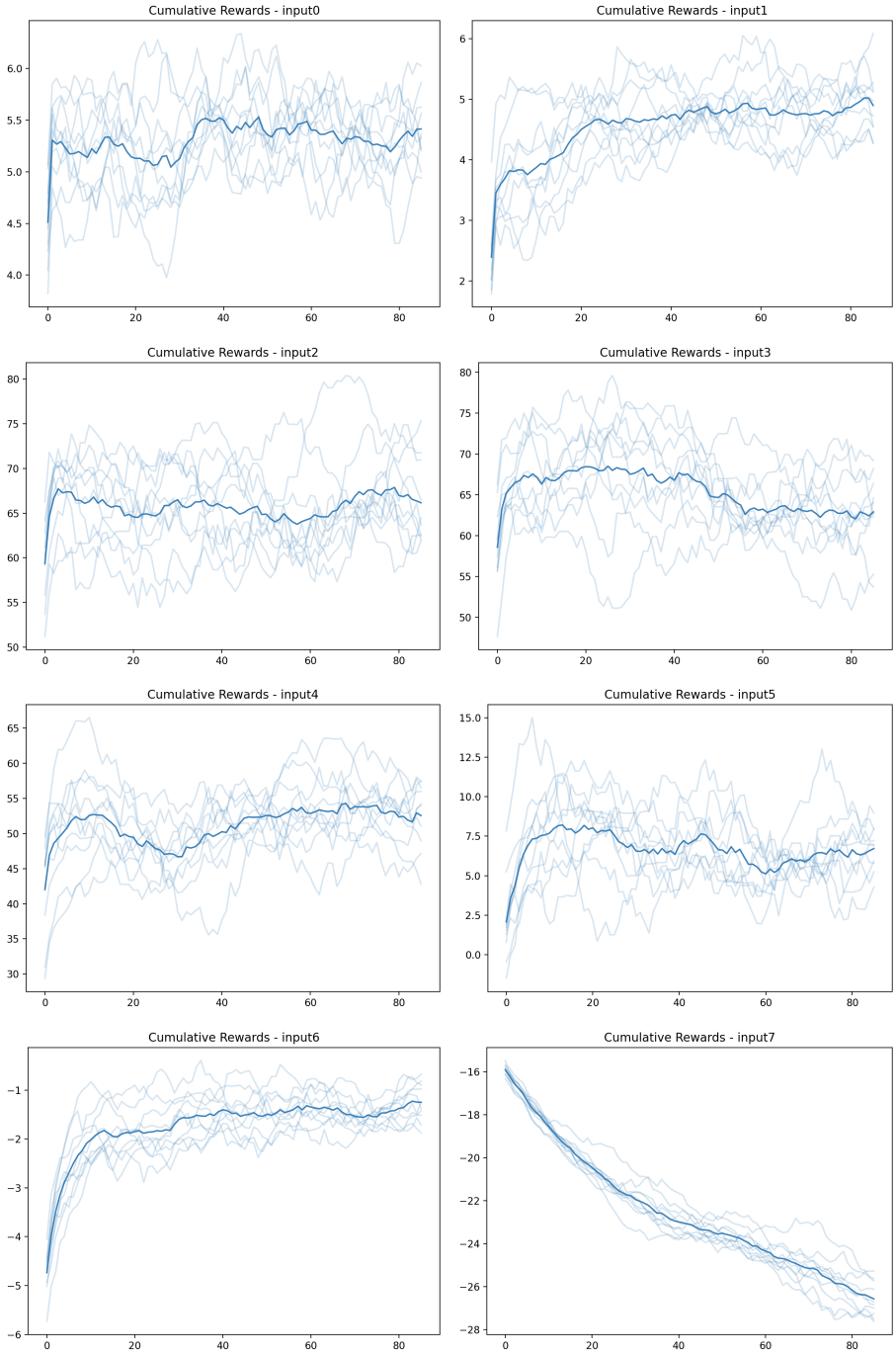

Figure 9: Resultados de *Learning Rate*, *Discount Factor* e *Exploration Factor*