**Resumo**

Calamidades sempre ocorreram no mundo e, muitas vezes, a improbabilidade de ocorrerem, as torna de difícil previsão e precaução. Sendo assim, busca-se o melhor comportamento e planejamento diante de um desastre, de forma a maximizar a quantidade de sobreviventes e minimizar o tempo de exposição ao perigo pelo qual as vítimas estão sendo submetidas. O objetivo deste trabalho é otimizar o planejamento do resgate de vítimas, dado um cenário em que conhecemos as prioridades de cada subespaço em nosso espaço de busca. Sendo assim, será utilizada o método Q Learning na área de *Reinforcement* *Learning* para a otimização da rota.

Além do principal objetivo e da aplicabilidade direta na área de calamidades, o objetivo é que a solução proposta possa ser aplicada a qualquer outro tipo de problema que envolva o deslocamento de um agente em um espaço condicionado por pontos previamente estabelecidos como prioritários com menor quantidade de passos.

**Problema e Premissas**

Cada subespaço do nosso espaço de busca será subdividido em quadrantes, onde cada quadrante terá uma prioridade pré-definida. Essa prioridade pode ser estabelecida de acordo com: maior chance de sobrevivência das vítimas contidas no quadrante, maior quantidade de vítimas no quadrante, ou qualquer outra forma de priorização. Uma possibilidade de obtenção da matriz é a utilização de um modelo de rede neural convolucional a partir de imagens de satélite para classificar determinado quadrante em relação a sua prioridade. Essa solução seria complementar ao trabalho aqui apresentado, que parte do resultado obtido como premissa. O objetivo do problema é identificar a melhor rota, de forma a maximizar as recompensas obtidas com menor quantidade de passos.

O matriz referente à premissa descrita acima segue conforme a imagem abaixo:

‘IMAGEM’

O problema será resolvido em etapas, da mais simples para a mais complexa. Sendo assim serão considerados os cenários

1. ‘muito simples’ (matriz 4x3)
2. ‘matriz simples’ (5x5)
3. ‘matriz média’ (8x8)
4. matriz complexa ‘9x13’.

A priorização segue conforme imagens destacadas abaixo.

**Modelagem**

O objetivo de cada jogo é percorrer todos os quadrantes com valores maiores que 0, com menor quantidades de passos e priorizando as maiores prioridades. Sendo assim foi definida variável deflatora d\_f, que é reponsável por reduzir as recompensas futuras de acordo com a quantidade de passos dados. O gráfico a seguir faz referência ao valor da variável ao termo de deflação em função da quantidade de passos dados

(Gráfico com curva com variável deflatora em cada curva é uma cor para cada valor estabelecido)

Para a matriz simples e matriz muito simples, foi tivemos o modelo rodado 10 vezes para cada configuração, sendo testados os seguintes hiperparâmetros:

* Learning rate = [0.01, 0.001, 0.0001]
* Gamma = [0.95, 0.99]
* d\_f = [0.8, 0.9, 0.95, 0.999]
* max exp\_rate (epsilon) = 1.0
* min exp\_rate decay (epsilon) = 0.01
* exp\_rate decay (epsilon) = [0.001, 0.005]
* quantidade maxima de passos = 5000
* quantidade de episódios = 8000

A atualização da tabela Q é feita a cada passo de acordo com a modelagem clássica:

(equação de atualização do q learning)

Além disso, podemos destacar que a ação é escolhida conforme épsilon atual, que segue a seguinte função para cada episódio. Quanto menor o épsilon, maior a chance de agente escolher ação que o traga melhores recompensas e quanto maior o épsilon, maior a chance de agente escolher ação aleatória.

Para os demais cenários, de acordo com o resultado obtido nos cenários i e ii, foi avaliado qual seria melhor opção para esses cenários, em que modelo foi rodado uma única vez (AVALIAR AQUI SE DE FATO FAREI DESSA FORMA)

**Resultado**

Os resultados podem ser divididos para cada problema estudado. Para os cenários i e ii , foi avaliado melhores recompensas obtidas no final do treinamento para cada cenário estudado conforme busca realizada conforme descrito acima

1. **Resultado Matriz Muito Simples (4x3)**

(Ordenação com melhores configurações)

(Gráfico comparativo com melhores configurações e intervalo de confiança)

(Giff do jogo jogado conforme matriz v table obtida)

(solução e recompensa obtida para resultado)

1. **Resultado Matriz Simples (5x5)**

(Ordenação com melhores configurações)

(Gráfico comparativo com melhores configurações e intervalo de confiança)

(Giff do jogo jogado conforme matriz v table obtida)

(solução e recompensa obtida para resultado)

1. **Resultado Matriz Média Complexidade (8x8)**

(Ordenação com melhores configurações)

(Gráfico comparativo com melhores configurações e intervalo de confiança)

(Giff do jogo jogado conforme matriz v table obtida)

(solução e recompensa obtida para resultado)

1. **Resultado Matriz Complexa (13x9)**

(Ordenação com melhores configurações)

(Gráfico comparativo com melhores configurações e intervalo de confiança)

(Giff do jogo jogado conforme matriz v table obtida)

(solução e recompensa obtida para resultado)

Conclusão

O resultado foi obtido com sucesso, para os cenários apresentados. Para futuros trabalhos, é interessante utilizar ténicas de deep reinforcement learning para