Inteligencia artifcial miniproyecto 3

Jose Gabriel Alvarez Medina* and Germán David Plazas Cayachoa†
Universidad del Rosario, Escuela de Ingeniería, Ciencia y Tecnología, Bogotá, Colombia
(Dated: Noviembre 2023)

I. PRESENTACIÓN DE LOS PROBLEMAS

En el ámbito de la inteligencia artificial, los entornos son una herramienta poderosa para evaluar algoritmos, métodos y técnicas. Cada entorno es único y presenta un reto diferente, dependiendo de su objetivo, condiciones, comportamiento y métricas. Por eso, para evaluar la eficacia de un algoritmo, no basta con evaluarlo en un único entorno. Es necesario evaluarlo en varios entornos para medir su robustez de una forma más polivalente.

En esta tarea, evaluaremos cuatro entornos diferentes, tres de los cuales se encuentran en la documentación de Gymnasyum. Es importante destacar que estos entornos guardan una estrecha relación con los de Gym, un proyecto desarrollado por OpenAI. OpenAI es una empresa líder en el desarrollo de herramientas de inteligencia artificial, como los modelos de procesamiento de lenguaje natural GPT, que se utilizan ampliamente y están transformando la modernidad, o los modelos de difusión para la generación de imágenes, como DALL-E 2.

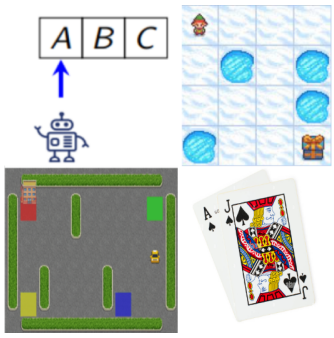


Figure 1. Entornos.

A. ABC

Nos ecntramos en un entorno que consta de tres casillas consecutivas: A, B y C, inicialmente me ubico en la casilla A y tengo la opción de moverme hacia la izquierda o la derecha donde mi objetivo principal es llegar a la casilla C, donde recibimos una recompensa de +10, mientras que moverme a cualquier otra casilla resulta en una recompensa de -1,Si decido moverme hacia la izquierda, el traslado es seguro, pero si me encuentro en la casilla A y elijo esta dirección, no me desplazo, si opto por moverme a la derecha, hay una probabilidad del 0.9 de desplazarme a la casilla de la derecha y una probabilidad del 0.1 de permanecer en la misma casilla.

Esta variabilidad añade un componente estocastico y estratégico donde debere considerar el riesgo de mis decisiones mientras busco desarrollar una estrategia que maximice las recompensas, evaluando cuidadosamente las opciones de movimiento y manejando la incertidumbre asociada con ciertas acciones.

B. Frozen lake

En este entorno, nos ponemos en la piel de un explorador que busca un legendario regalo. Sin embargo, durante su búsqueda, queda atrapado en un lago congelado. Esta es una situación difícil, ya que el explorador corre el riesgo de caer en la parte no congelada del lago y tener un frio final.

El entorno consiste en una cuadrícula de 4x4 casillas, en la que el explorador puede moverse en las cuatro direcciones cardinales. Cuando el explorador llega a una frontera del lago, rebota. Las acciones de moverse o caerse no tienen recompensa, pero si el explorador llega al regalo, recibe una recompensa de +1.

El piso del lago es resbaladizo, lo que hace que el explorador se deslice con facilidad. Esto dificulta aún más su tarea de llegar al regalo sin caerse en el agua.

* Correspondence email address: josega.alvarez@urosario.edu.co
† Correspondence email address: germand.plazas@urosario.edu.co

C. Taxi

En este entorno, nos ponemos en la piel de un taxista. Nuestro objetivo es recoger a un pasajero en su ubicación y llevarlo a su destino.

El entorno consiste en una cuadrícula de 6x6 casillas. Podemos movernos en las cuatro direcciones cardinales, recoger y dejar pasajeros, y empezar en una ubicación aleatoria. Las ubicaciones del punto de recogida y destino del pasajero pueden ser uno de los cuatro puntos representados por los colores azul, rojo, amarillo y verde.

Las recompensas para este entorno son las siguientes:

Por cada movimiento se resta 1 punto, una recogida o una dejada de pasajero errónea resta 10 puntos, dejar al pasajero en su destino correctamente suma 20 puntos.

D. Blackjack

En este juego de cartas, nos enfrentamos a un desafío contra el crupier y la fortuna, donde la combinación de habilidad y suerte es esencial, en el blackjack, el objetivo radica en sumar 21 para ganar la ronda, pero esto se complica, pues pasarse de dicho valor resulta en la derrota. Nos toca transitar por la fina línea de acercarnos al máximo sin fallar en el intento.

En este entorno, empezaremos con el crupier sacando cartas de un mazo infinito, dejando una de ellas boca arriba y la otra boca abajo. Como jugador, mis cartas estarán expuestas desde el principio. El valor de mis cartas se suma de la siguiente manera: J, Q y K valen 10, mientras que los ases pueden ser 1 o 11 según convenga, y las demás cartas suman su valor numérico.

Tengo la opción de solicitar cartas adicionales hasta que decida plantarme o hasta que supere el valor de 21, momento en el cual perderé. Si opto por plantarme, el crupier revelará la carta que estaba boca abajo y sacará cartas hasta alcanzar una suma igual o superior a 17. En caso de que el crupier se exceda, ganaré; de lo contrario, el ganador será quien se haya aproximado más a 21.

Las recompensas se distribuyen de la siguiente manera: +1 por ganar una ronda, -1 por perder, y en caso de empate, no sumaré ni restaré puntos. Sin embargo, si obtengo un blackjack con J, Q o K combinado con un as, la recompensa será de +1.5.

II. RESULTADOS Y DISCUSIÓN

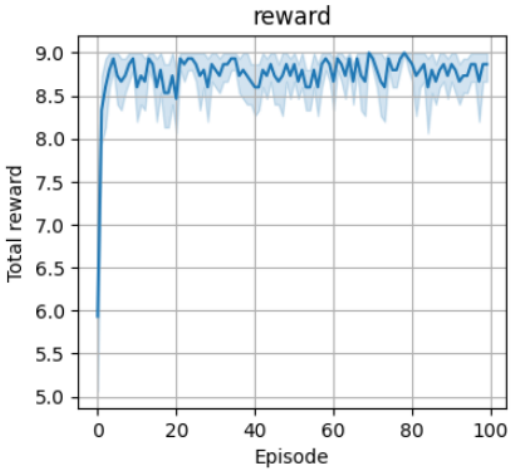
A. Entorno ABC

Para este entorno vamos a usar el algoritmo *SARSA* para el mejoramiento de la politica. Establecemos los parametros del experimento como se ven a continuación.

```
parameters = {\
    'nS': 3,\
    'nA': 2,\
    'gamma': 0.8,\
    'epsilon': 0.0001,\
    'alpha': 0.1, \
}

agent = [SARSA(parameters=parameters)]

# Create experiment
experiment = Experiment(environment=env,\
                 env_name='ABC', \
                 num_rounds=100, \
                 num_episodes=100, \
                 num_simulations=15)

# Use stored agents to run test
experiment.run_experiment(
                agents=agent,\
                names=['SARSA'], \
                measures=['reward', 'hist_reward'],\
                learn=True)
```

Figure 2. Parametros y experimento ABC.

Es importante resaltar el número de rondas 100, episodios 100 y simulaciones 15. Una vez se realiza el entrenamiento, se obtiene la recompensa total por episodio, como se observa en la siguiente imagen.




```
Average sum of rewards:
 model
SARSA    131.21
Name: reward, dtype: float64

Episode termination percentage:
 model
SARSA    1500.0
Name: done, dtype: float64
```

Figure 3. Imagen de recompensa total .

Ahora, el histograma correspondiente a las recompensas totales por episodio:

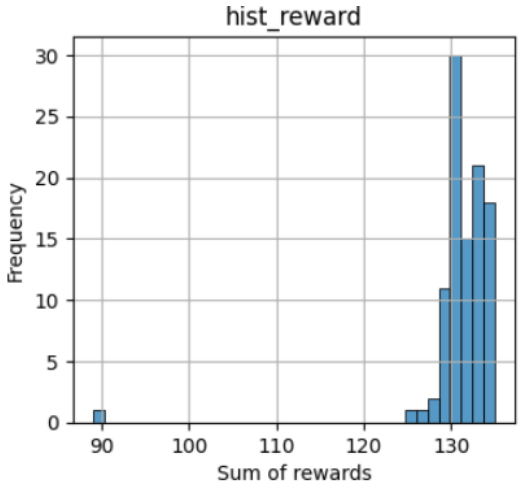
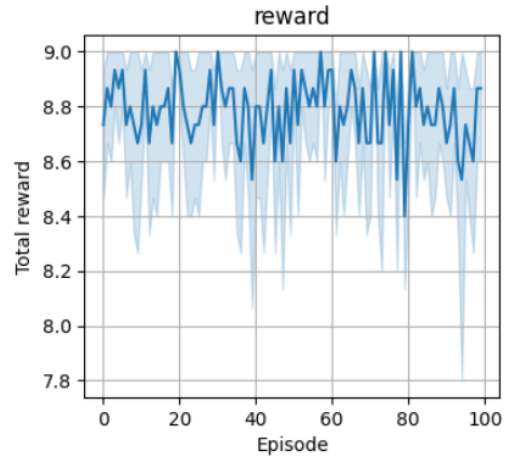


Figure 4. Histograma recompensa total.

Es de notar que se resuelve el entorno pues se obtiene una recompensa total promedio de 131.8 en el test.




```
Average sum of rewards:
 model
SARSA    131.8
Name: reward, dtype: float64
```

Figure 5. Imagen de recompensa total test.

Finalmente, la política ótpima y el valor q para cada estado se pueden ver en la siguiente imagen:

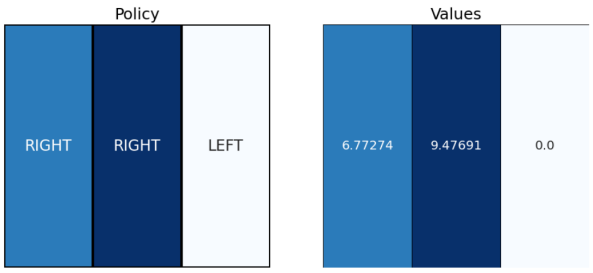


Figure 6. Política y valores q.

Se puede concluir que el algoritmo de SARSA ha ayudado a que le agente aprenda una política óptima para el entorno.

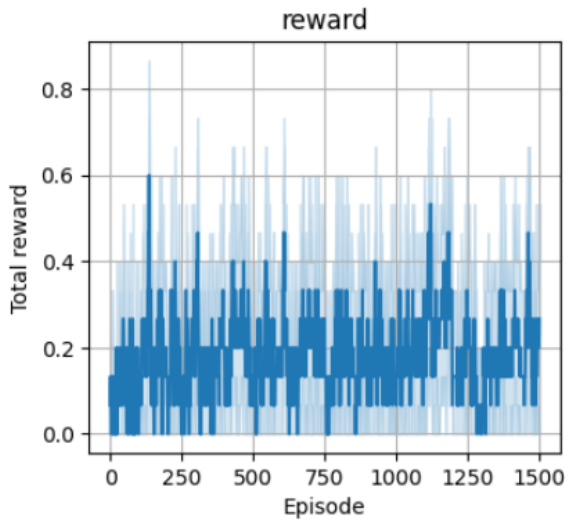
B. FrozenLake Para este entorno se tienen en cuent a los parametros de entorno *desc, is_slippery* con respectivos valores de [”SFFF”,”FFHF”, ”FHFF”,”FFFG”], True. Se desea utilizar el algoritmo de mejoramiento de politica Q-learning. Los paramétros de entrenamiento del entorno son

```
parameters = {\
    'nS': size*size,\
    'nA': env.action_space.n,\
    'gamma': 0.99,\
    'epsilon': 0,\
    'alpha': 0.1, \
}

agent = [Q_learning(parameters=parameters)]

experiment = Experiment(environment=env,\
                env_name='FrozenLake', \
                num_rounds=100, \
                num_episodes=1500, \
                num_simulations=15, \
                state_interpreter=gym_interpreter1)
```

Figure 7. Parametros de entrenamiento FrozenLake.

Es de notar que $\gamma = 0.99, \epsilon = 0, \alpha = 0.1$. Durante el proceso de aprendizaje, se quiere que el agente pruebe distintos cursos de acción, de manera tal que tenga una mayor confianza en que está llegando a una política óptima. No obstante, a la hora de poner a marchar al agente en producción, se desea que el agente tenga su mejor desempeño. Por esto, el parámetro ϵ es igualado a cero.

La recompensa total por episodio se puede ver en la siguiente imagen




```
Average sum of rewards:
 model
Q_learning    2.678667
Name: reward, dtype: float64

Episode termination percentage:
 model
Q_learning    1482.733333
Name: done, dtype: float64
```

Figure 8. Imagen de recompensa total.

El histograma de recompensa total es el siguiente

Figure 9. Histograma recompensa total.

Y finalmente, la pólitica óptima y el valor q óptimo de cada estado viene dado por:

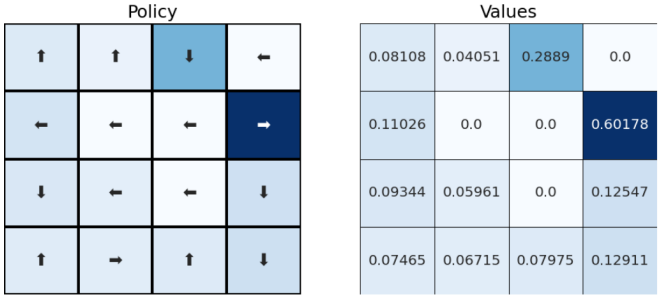


Figure 10. Política y valores q.

Nótese que si se aumenta el número de episodios a realizar, el valor q para cada estado va a ser más preciso, lo mismo ocurrirá con la política, a mayor numero de episodios, la política será más óptima.

C. Entorno Taxi

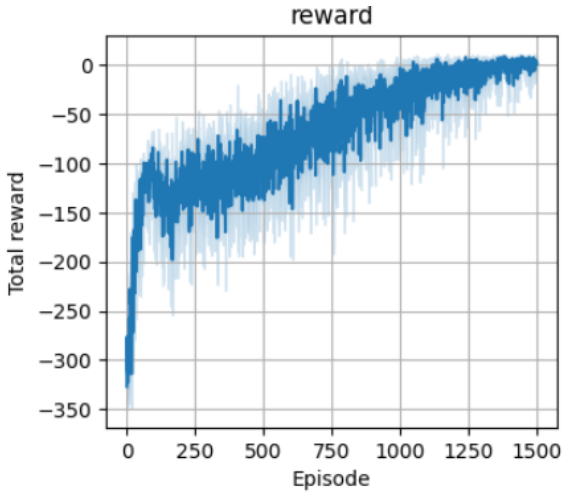
Para resolver este entorno se hace uso del algoritmo Q-learning. Se establecen los siguientes parámetros.

```
size=500
# Create agent
parameters = {
    "nS": size*size,
    "nA": env.action_space.n,
    "gamma":0.99,
    "epsilon":0.0001,
    'alpha': 0.1
}


#Entrenamiento del agente
agent = [Q_learning(parameters=parameters)]

experiment = Experiment(environment=env,\
                env_name='Taxi', \
                num_rounds=100, \
                num_episodes=1500, \
                num_simulations=10, \
                state_interpreter=gym_interpreter1)
```

Figure 11. Parametros de entrenamiento Taxi.

Se tienen en cuenta los mismos valores de γ, α, ϵ que en el entrono anterior $\gamma = 0.99, \epsilon = 0, \alpha = 0.1$.

La recompensa total por episodio se puede ver en la siguiente imagen.




```
Average sum of rewards:
 model
Q_learning   -669.162
Name: reward, dtype: float64

Episode termination percentage:
 model
Q_learning    661.466667
Name: done, dtype: float64
```

Figure 12. Imagen de recompensa total.

El histograma de recompensa total es el siguiente

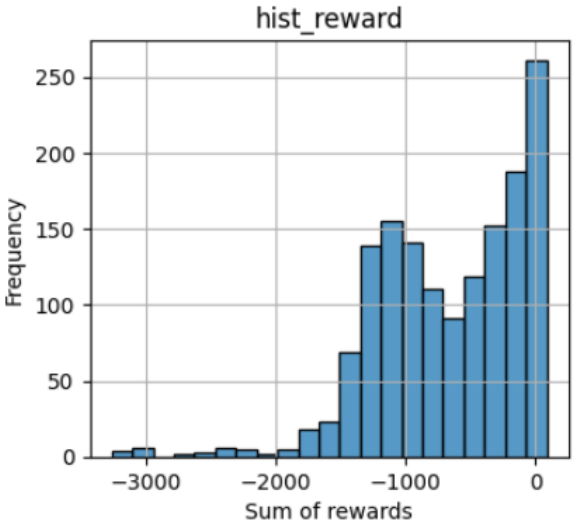


Figure 13. Histograma recompensa total.

Y finalmente, la pólitica óptima y el valor q óptimo de cada estado viene dado por:

```
Valores Q de cada estado:

[[ 0.          0.          0.          0.          0.          0.        ]
 [-2.71995307 -2.66519075 -2.7599249  -2.66519075 -2.70180328 -2.997001  ]
 [-2.05293037 -1.97348051 -2.05330698 -1.98111352 -2.00779563 -1.999     ]
 ...
 [ 0.          0.          0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.          0.          0.        ]
 [ 0.          0.          0.          0.          0.          0.        ]]
250000
```

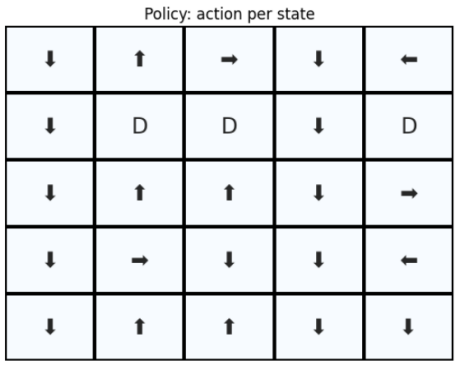
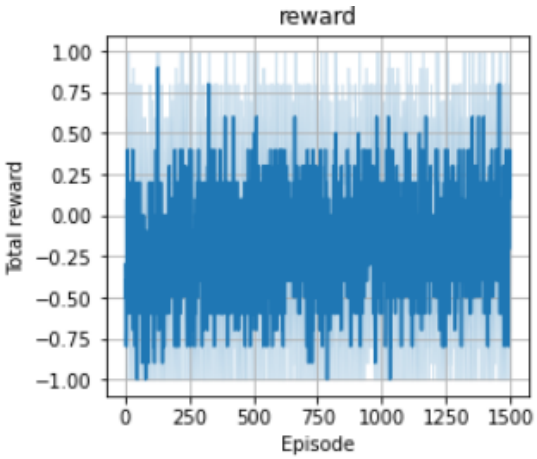


Figure 14. Política y valores q.

En este caso, el número de episodios, rondas y simulaciones jugan un papel muy importante pues hay 500 estados discretos, ya que hay 25 posiciones de taxi, 5 posibles ubicaciones del pasajero (incluido el caso en que el pasajero está en el taxi) y 4 ubicaciones de destino.

D. BlackJack

La recompensa total por episodio se puede ver en la siguiente imagen




```
Average sum of rewards:
 model
Q_learning   -1.760667
Name: reward, dtype: float64
```

Figure 15. Imagen de recompensa total.

El histograma de recompensa total es el siguiente

Figure 16. Histograma recompensa total.

III. RESULTADOS Y DISCUSIÓN