Aplicación de Q-Learning & Deep Q-Learning a la resolución de problemas

Abad L. Freddy L.*, Cabrera C. Edwin F.*, Campoverde C. Daniel A.*.
Carrera de Ingeniería de Sistemas.
Facultad de Ingeniería.
Universidad de Cuenca.

Objetivo General

Emplear los algoritmos Q-Learning y Deep Q-Learning en la resolución del problema “La Final”, un pequeño juego donde un arquero de fútbol aprende, mediante aprendizaje reforzado, a atajar tiros de penal.

Objetivos específicos

1. Desarrollar los conocimientos de Aprendizaje Reforzado en un problema puntual de dificil resolución.
2. Observar la curva de aprendizaje aplicando el algoritmo de Q-Learning en el juego ”La Final”.
3. Observar la curva de aprendizaje aplicando el algoritmo de Deep Q-Learning en el juego ”La Final”.
4. Contrastar resultados de los algoritmos de Q-Learning & Deep Q-Learning frente a la resolucion del problema resuelto.
5. Utilizar los conocimientos de programación aplicada al Machine Learning para la resolución de problemas del diario vivir

Antecedentes

El aprendizaje reforzado es una área de machine learning inspirada en la psicología del comportamiento, en esta se modela el entorno como un proceso de decisión de Markov (MDP), donde ciertas acciones tomadas por el agente modifican el estado y suponen una cierta recompensa. El aprendizaje supervisado es particularmente apropiado para problemas que involucran beneficios a corto y largo plazo donde el beneficio total obtenido de ambos a de ser maximizado.
Q-Learning es una técnica de aprendizaje reforzado de tipo model-free, es decir, no requiere que el entorno sea previamente formulado en un modelo. Puede ser empleado para encontrar reglas de acción óptimas para un proceso de decisión de Markov aprendiendo una función $Q(s, a)$ que representa el beneficio de tomar una acción a en un estado s.
El uso de técnicas de aprendizaje reforzado permite crear soluciones para tareas específicas, sin embargo, es altamente deseable encontrar soluciones generalizadas que sean aplicables en varias tareas diversas.
Con este objetivo, Deep Q-Learning combina Q-Learning con aprendizaje profundo, empleando estas para representar la tabla Q. Para ello; se construye una red neuronal convolucional que puede ser aplicada en diferentes entornos. Un entorno se compone del actor (red neuronal en el entorno), un conjunto de movimientos posibles, y las penalizaciones y recompensas.

El Q-Learning involucra a un agente, un conjunto de estados *S*, y un conjunto *A* de acciones por estado. Llevando a cabo una acción $a \in A$, el agente pasa de un estado a otro $a \in S$. La ejecución de una acción en un estado concreto le proporciona una recompensa al agente (una puntuación numérica).
El objetivo del agente es maximizar su (futura) recompensa total. Lo hace sumando la máxima recompensa alcanzable, en estados futuros, a la recompensa por alcanzar su estado actual, influyendo eficazmente en la acción actual por la futura recompensa potencial. Esta recompensa potencial es una suma ponderada de la esperanza matemática de las recompensas de los futuros pasos empezando desde el estado actual.
Google DeepMind creó una red neuronal (usando Deep Q-Learning) que aprende cómo jugar a los videojuegos de una manera similar a la de los seres humanos, utilizó redes neuronales convolucionales profundas, con capas de filtros convolucionados enladrillados para asemejar los efectos de los campos receptivos. El aprendizaje por refuerzo es inestable o divergente cuando un aproximador de funciones no lineales se utiliza para representar *Q*. Esta inestabilidad proviene de la correlación presente en las secuencias de observación, el hecho de que actualizaciones pequeñas de *Q* puedan cambiar significativamente la política y la distribución de los datos, y las correlaciones entre *Q* y los valores objetivo.

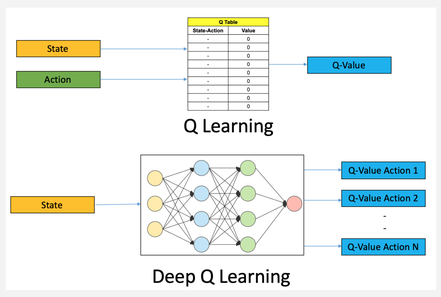

Figura 1: Q-Learning & Deep Q-Learning

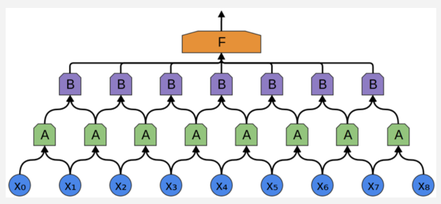

Figura 2: Estructura red neuronal convolucional

Figura 3: Alpha GO - Google DeepMind

Metodología

Desarrollamos la solución al problema planteado usando el lenguaje python y la librería pygame para el control del aspecto gráfico del juego.
El juego emplea la división de cuadrículas para el desplazamiento de los elementos, según la especificación, la interfaz resultante se observa en la Figura 4.

Figura 4: Acciones del Juego

La solución se estructura en 3 clases, según se observa en la Figura 5. El estado de las instancias correspondientes soporta el algoritmo de aprendizaje y el control gráfico del juego, proceso que se ejecuta en un bucle de episodios.

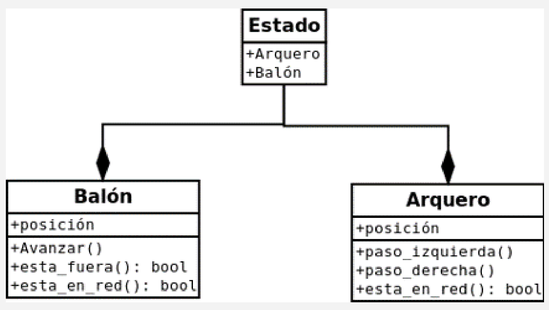

Figura 5: Diagrama de Clases

El funcionamiento general a alto nivel de la solución se aprecia en la Figura 6; Existe una matriz Q de dimensiones $S\ x\ A$, donde S es el número de estados posibles en el juego y A el número de acciones posibles. Cada elemento de la matriz Q representa el beneficio de tomar una acción a desde cierto estado s.

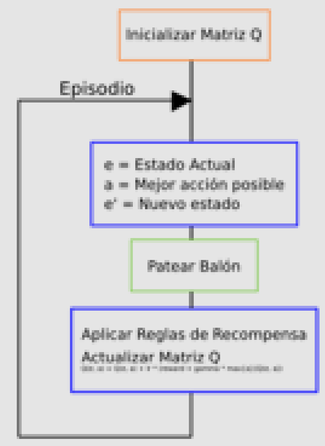

Figura 6: Funcionamiento de alto nivel.

Resultados

Figura 7: Interfaz del Juego.

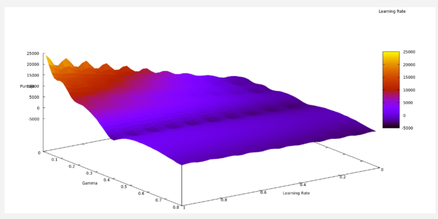

Figura 8: Superficie de optimización Q-Learning.

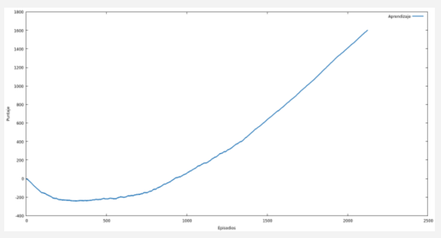

Figura 9: Curva de aprendizaje Q-Learning - Número de Penales vs. Puntaje Acumulado.

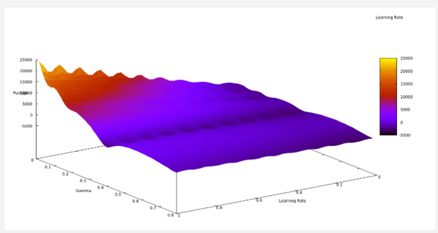

Figura 10: Superficie de optimización Deep Q-Learning.

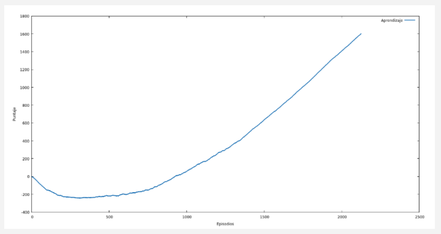

Figura 11: Curva de aprendizaje Deep Q-Learning - Número de Penales vs. Puntaje Acumulado.

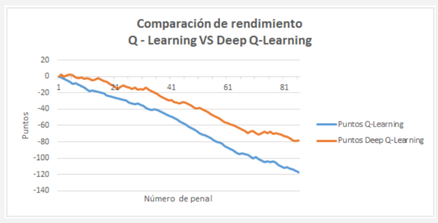

Figura 12: Comparación del Rendimiento Q-Learning vs.Deep Q-Learning.

Prueba	Penales	Goles	Fuera	Tapadas	Puntos
1	60	32	27	1	-55
2	60	31	27	2	-49
3	60	29	29	2	-53
4	60	28	27	5	-43
5	60	31	24	5	-48
6	60	27	30	3	-34
7	60	32	25	3	-57
8	60	30	27	3	-53
9	60	32	22	6	-46
10	60	30	25	5	-41
Media	**60**	**30**	**26**	**4**	**-48**

Figura 13: Rendimiento Deep Q-Learning.

Conclusiones

Se emplearon los algoritmos Q-Learning y Deep Q-Learning para la resolución del problema planteado, y se estudiaron las configuraciones de sus parámetros influyentes, permitiéndonos obtener el máximo aprendizaje posible por episodio. Descubrimos que la configuración óptima para el problema resuelto ocurre con los parámetros $lr = 1; Gamma = 0$, la misma que logra un comportamiento libre de errores en aproximadamente 1500 episodios.

Dado el comportamiento aprendido para el número de episodios de entrenamiento requerido, consideramos que los algoritmos empleados ofrecen un excelente rendimiento y aplicabilidad al problema resuelto.

Líneas/Proyecto de Investigación

- **Línea de investigación #1:** Recopilación de información necesaria y la creación del algoritmo de aprendizaje por refuerzo.
- **Línea de investigación #2:** Diseño de diagramas de clase para programar las clases que intervienen en el problema.
- **Línea de investigación #3:** Proceso de recopilacion de datos, segun un numero de épocas (pasadas) que mejoren los resultados
- **Línea de investigación #4:** Finalmente, se comprueba la aplicación a partir de las épocas que mejores resultados se obtuvieron.

Información de contacto

- Email:{freddy.abadl, edwin.cabrera, daniel.campoverde}@ucuenca.edu.ec