Trabajo Práctico 2 - Aprendizaje por Refuerzo

Inteligencia Artificial y Neurociencias

Segundo semestre de 2025

El objetivo de este trabajo práctico es construir un agente capaz de jugar al juego "Connect 4", también conocido como "Cuatro en línea".

Las **reglas del juego** están explicadas en detalle en `https://www.casualarena.com/es/conecta-4/reglas`.

Se pide implementar un agente de aprendizaje por refuerzo que utilice alguno de los algoritmos vistos en clase (por ejemplo, *Deep Q-Learning*) y un ambiente en el cual el agente puede entrenarse. Esto incluye definir la noción de estado y ambiente del juego.

Se proveen los siguientes archivos:

- `connect4.py`: Clase `Connect4`, que permite jugar un juego completo a un `Agente`.
- `agentes.py`: Clase abstracta `Agente` y tres implementaciones de muestra: `RandomAgent`, que elige sus jugadas al azar, `HumanAgent`, que permite al usuario jugar por consola y `DefenderAgent`, que revisa si el oponente está por ganar e intenta bloquearlo; si no, juega al azar.
- `principal.py`: Tiene una propuesta de clases y métodos que podrían implementar como solución. Es simplemente una guía, no es necesario usarla o completarla.
- `entrenar.py`: Script para entrenar y almacenar la política de un agente siguiendo la interfaz presentada en `principal.py`.
- `jugar_humano_contra_defensor.py`: Script que carga un `HumanAgent` y un `DefenderAgent` y ejecuta un juego completo de Connect4. Una vez hayan creado su `TrainedAgent`, pueden usarlo para hacerlo jugar contra cualquiera de los agentes que se encuentran en el archivo `agentes.py` y así evaluar su rendimiento.
- `utils.py`: Funciones y constantes útiles a todo el proyecto.

Este TP debe realizarse en grupos de **dos (2) a cuatro (4) integrantes**. Será puntuado con una nota binaria, de 0 puntos, o bien de 100 puntos. **La entrega no es obligatoria**; quienes no entreguen tendrán nota 0. (Recomendamos revisar las reglas de aprobación de la materia.)

Condiciones para lograr 100 puntos en este TP:

1. Entrenar un agente con uno de los algoritmos de Aprendizaje por Refuerzo vistos en clase (recomendamos intentar con *Deep Q-Learning*, pero pueden usar otro: *Monte Carlo*, *Q-Learning*, etc.).

2. Con la política aprendida, implementar y entregar un `TrainedAgent` que implemente la clase abstracta `Agent` y que se ejecute correctamente en la clase `Connect4`.

3. Presentar todos los archivos de código, junto a un breve informe (máximo dos carillas) detallando las decisiones tomadas, el grado de éxito que tuvieron y cualquier otro comentario que consideren relevante para entender el trabajo realizado.

4. La condición mínima de aprobación requiere que el `TrainedAgent` entregado supere la gran mayoría de las veces al `RandomAgent`.

Fecha límite de entrega: viernes 3/10 a las 23:59hs. Este TP no tiene recuperatorio.

Comentarios adicionales:

- La mayor dificultad del problema pasa por el diseño del ambiente (definir los estados, las acciones, las recompensas, etc.) y del agente (ajustar los parámetros ϵ, γ, α, definir la arquitectura de la red si se usa DQN, etc.). Esas cosas son mucho más desafiantes (y más interesantes) que la escritura del algoritmo de RL en sí misma.

- Como el Connect 4 se juega de a dos, es parte del desafío saber con qué oponente/s entrenar al agente. Para esto, sugerimos explorar más opciones además de la propuesta en el archivo `entrenar.py`.

- En caso de usar DQN, experimentar con diferentes hiperparámetros y arquitecturas de la red hasta hallar un agente lo suficientemente bueno a la hora de jugar al Connect4.

- **Torneo:** Con todos los agentes entregados, haremos un campeonato para elegir a los mejores jugadores de Connect4!

- Un baseline más exigente para el `TrainedAgent` que entreguen, será que supere ampliamente al `DefenderAgent`.

- Si encuentran bugs en nuestro código, por favor avisar.