Deep Reinforcement Learning aplicado a Super Mario Bros

Sistemas Multirobot - Curso 2024/2025

1 Introducción

En esta práctica, vais a implementar un agente con Aprendizaje por refuerzo profundo (Deep RL) para jugar una versión clásica del videojuego Super Mario Bros.

Figure 1: Super Mario Bros en NES

Para ello, vamos a utilizar la librería **Gymnasium**, que es el estándar en diseño de entornos de entrenamiento para RL. Esta librería nos permite definir los diferentes componentes del entorno (observaciones, transiciones, función de recompensa, etc.) e interactuar con este de forma más ordenada. Podéis consultar la documentación en su web `https://gymnasium.farama.org/`

En cuanto al entorno, vamos a utilizar un entorno base ya diseñado y que sigue la API de Gymnasium. Este entorno utiliza un emulador de NES para Python, y permite jugar a todos los niveles del juego original. Podéis consultar toda la documentación del entorno en el siguiente enlace: `https://github.com/Kautenja/gym-super-mario-bros`.

En el repositorio tenéis explicado en detalle como se define el entorno, como se pueden iniciar entornos de los diferentes niveles, cómo se calcula la recompensa por defecto, y la información adicional que devuelve el emulador tras cada 'step' del entorno (cada iteración del bucle clásico del RL). **Se recomienda leer en detalle esta información antes de realizar la práctica**.

En cuanto a los algoritmos de entrenamiento, vamos a utilizar la librería **Stable Baselines 3**. Esta librería es la más accesible al público menos experto, y cuenta con bastantes algoritmos implementados, entre ellos algunos de los vistos en clase como DQN, PPO o SAC. La web de la librería cuenta con muchos ejemplos, una documentación extensiva, y enlaces a materiales

adicionales. Podéis consultarla en este enlace: `https://stable-baselines3.readthedocs.io/en/master/index.html`.

2 Objetivos

El objetivo de la práctica es que os familiaricéis con la terminología del campo de RL, y que os enfrentéis a un reto más complejo que los ejemplos clásicos que se suelen ver al empezar a estudiar estos conceptos. Sin embargo, no dudéis en estudiar en detalle dichos ejemplos (Cartpole, LunarLander, etc.) para mejorar vuestro entendimiento de los conceptos base y su implementación.

Para la práctica en sí, se pretende que partais de la implementación inicial del entorno proporcionada, la ampliéis/modifiquéis según creáis conveniente, y finalmente entrenéis un agente cuyo desempeño sea lo mejor posible.

Cómo objetivos mínimos, los estudiantes deberán ser capaces de:

- Entender la API de Gymnasium, sus partes principales, y la implementación concreta del entorno de Super Mario Bros
- Modificar componentes clave del entorno proporcionado, tales como las observaciones proporcionadas al agente o la función de recompensa.
- Implementar un proceso de aprendizaje usando Stable Baselines 3, escogiendo un algoritmo de forma justificada, y demostrando un rendimiento óptimo del agente.
- Analizar el proceso de ampliación del entorno y diseño del entrenamiento formalmente, justificando las decisiones tomadas en la memoria asociada a la práctica.

Además, con esta práctica se pretende que seáis autónomos en cuanto a búsqueda de recursos y toma de decisiones. Se os proporciona un código base con indicaciones y recomendaciones, pero se os insta a **investigar, probar, y ampliar todo lo que consideréis**. No hay una única forma de resolver la práctica, así que se tendrá en cuenta la iniciativa, creatividad y resolutividad del alumno como parte del proceso de evaluación.

3 Setup Inicial

Como punto de partida de la práctica, se os proporciona una carpeta comprimida con los siguientes ficheros:

- *requirements.txt*: Archivo con las dependencias necesarias para ejecutar correctamente el código proporcionado.
- *mario_rl.ipynb*: Jupyter Notebook con el código plantilla, así como algunas guías y explicaciones para ayudaros en la implementación.

Además, **es obligatorio el uso de Python 3.8**. Para ello, cread un entorno de Conda/Virtualenv para la práctica, e instalad los paquetes requeridos dentro del entorno:

```
conda create -n mario_env python=3.8
conda activate mario_env
cd /path-to-ipynb-file
pip install -r requirements.txt
```

4 Tareas a realizar

4.1 Configuración del entorno y exploración

1. Inicializar el entorno por defecto
2. Implementar un bucle tomando acciones aleatorias y visualizar los resultados
3. Familiarizaros con los espacios de observaciones y acciones.

4.2 Modificación del entorno inicial

1. Preprocesar la observación devuelta por el entorno
2. Ampliar la función de recompensa base
3. Implementar una función de creación de nuevos entornos con las adiciones implementadas.

4.3 Entrenar un agente usando StableBaselines3

1. Investigar la librería Stable Baselines 3 para entender su funcionamiento
2. Escoger la arquitectura y algoritmo adecuados para nuestro entorno.
3. Entrenar el agente y monitorizar el proceso de aprendizaje.

4.4 Evaluación del agente entrenado

1. Evaluar el agente resultante del entrenamiento con métricas objetivas
2. Visualizar el comportamiento del agente
3. Analizar los resultados obtenidos e iterar hasta conseguir un comportamiento óptimo.

5 Instrucciones para la entrega

La entrega consistirá en un único archivo comprimido ZIP que contendrá:

- Vuestro Jupyter Notebook con la implementación completa, así como los outputs de cada celda una vez ejecutada en su versión final.

- Un informe en PDF explicando cada parte de vuestra implementación, justificando las decisiones tomadas (preprocesado aplicado, algoritmo escogido, etc.), y métricas del agente final.
- Archivo .zip de vuestro modelo definitivo entrenado con Stable Baselines 3 (resultado de 'model.save(path-to-model)'.

6 Recomendaciones e información adicional

6.1 Sobre los recursos de cómputo:

- Siempre que sea posible, intentad ejecutar el código localmente en vuestro PC. Esto os dará mucho más control sobre la instalación y las versiones utilizadas.
- Aquellos de vosotros cuyo PC no tenga las prestaciones suficientes, usad Google Colab para ejecutar el Notebook en la nube y hacer uso de hardware más potente. Se ha preparado la práctica en un Jupyter Notebook para facilitar su uso a aquellos que lo necesiten.
- Si aun así tenéis problemas, hacedlo saber al profesorado y trataremos de buscar una solución.

6.2 Sobre el tiempo de entrenamiento:

- **Es normal que los entrenamientos tarden bastante tiempo**, como en cualquier método de aprendizaje automático. Podéis esperar mínimo 1h por cada intento de aprendizaje para poder empezar a observar comportamientos satisfactorios.
- El tiempo de entrenamiento depende de muchos factores: Hardware empleado, tipo de preprocesado que apliquéis, número de 'steps' del entreno... Verificad primero que todo está como esperais (preprocesado correcto, espacio de acciones deseado, etc.), y entonces lanzad entrenos largos.
- Aprovechad el tiempo mientras el agente está entrenando: Revisad vuestra implementación, pensad mejoras, documentad vuestros experimentos, etc.
- Cuando creais tener una buena configuración, aumentad considerablemente el tiempo de entrenamiento, dejando el ordenador entrenando durante varias horas (mientras coméis, trabajais en otras asignaturas, durante la noche...).

6.3 Uso de ChatGPT o similares:

- **Está permitido el uso de LLMs para la práctica.** Consultad lo que queráis, usadlo para resolver errores, pedid consejo para ajuste de hiperparámetros, etc.
- **Indicad en la memoria dónde y para qué lo habéis usado.** No se penalizará su buen uso, es más, se recomienda. Sin embargo, se os pide que lo reflejéis en la memoria, ya que así podremos saber que si solucionáis algo de forma 'mágica', no os habéis copiado de nadie.
- **Sed críticos con los LLMs**. Son una herramienta potente, pero usadlos con criterio y no aceptéis cualquier respuesta sin analizarla y entenderla.

7 Criterio de evaluación

Para la evaluación de la práctica, se tendrá en cuenta no solo el desempeño del agente, sino la explicación y la justificación del proceso de toma de decisiones e implementación que haréis durante la práctica.

A grandes rasgos, la distribución aproximada de nota será la siguiente:

- Configuración inicial del entorno: 10%
- Modificación del entorno inicial: 40%
- Entrenamiento del agente con SB3: 40%
- Evaluación del agente entrenado: 10%

Todas las partes de la implementación, decisiones de diseño tomadas, análisis de resultados, iteración de diseño, etc. **debe estar debidamente documentado**. Si el agente tiene un buen rendimiento pero la documentación de la práctica es pobre, **la práctica se evaluará desfavorablemente**. El objetivo de esta práctica es que aprendáis y os acerquéis al campo del RL, así que se evaluará positivamente que demostréis entender lo que estáis haciendo y analizar por qué funciona o no vuestro agente.

8 Recursos

- Gym Super Mario Bros: `https://github.com/Kautenja/gym-super-mario-bros`
- Gymnasium: `https://gymnasium.farama.org/`
- Stable Baselines 3: `https://stable-baselines3.readthedocs.io/en/master/index.html`
- Google Colab: `https://colab.research.google.com/`