.UBAfiuba
FACULTAD DE INGENIERÍA

TP Final

Aprendizaje por Refuerzo

Juan Pablo Alianak

GitHub: https://github.com/jpalianak/rl2

Contenido

1. Introducción

El presente trabajo aborda el desarrollo de un agente inteligente para la toma de decisiones en operaciones de trading algorítmico mediante técnicas de aprendizaje por refuerzo profundo. En particular, se emplea un algoritmo Double Deep Q-Network (DQN) con una arquitectura de red neuronal basada en LSTM (Long Short-Term Memory) para capturar patrones temporales en series de precios históricos.

El agente se entrena y evalúa utilizando datos históricos reales de acciones bursátiles, simulando un entorno de inversión donde puede comprar, vender o mantener activos. Para ello, se implementa un entorno personalizado que permite representar las dinámicas de mercado y las restricciones operativas.

Este enfoque busca optimizar la estrategia de inversión maximizando las ganancias acumuladas a lo largo del tiempo, superando métodos tradicionales mediante el aprendizaje autónomo y adaptativo.

2. Descripción del entorno

Para el entrenamiento del agente se desarrolló un entorno personalizado. Esta decisión permitió mayor control sobre la lógica interna del ambiente.

El entorno simula un mercado financiero utilizando datos históricos diarios del activo bursátil. Cada observación es una ventana temporal de 20 días compuesta por cinco características: precio de apertura, máximo, mínimo, cierre y volumen. A partir de esta información, el agente debe decidir entre tres acciones posibles:

0: Mantener posición

1: Comprar una acción

2: Vender una acción

La recompensa en cada paso se calcula como la diferencia en el valor total del portafolio (acciones + efectivo) entre dos días consecutivos. El entorno también incluye restricciones como la imposibilidad de vender sin tener acciones en cartera.

3. Algoritmo de aprendizaje

El agente utiliza una arquitectura de aprendizaje por refuerzo profundo basada en Double Deep Q-Learning (DQN) con mejoras para estabilizar el entrenamiento:

- Red neuronal LSTM: Permite capturar la secuencia temporal de los datos y detectar patrones en la evolución de los precios.
- Replay Buffer: Almacena transiciones pasadas (estado, acción, recompensa, siguiente estado) para romper la correlación temporal y reutilizar experiencias.
- Target Network: Se actualiza periódicamente para proporcionar objetivos más estables al momento de actualizar la red principal.
- Política epsilon-greedy: Controla el equilibrio entre exploración y explotación. Se inicia con un alto valor de epsilon (exploración aleatoria) que decrece progresivamente para fomentar la explotación de la política aprendida.

Los hiperparámetros clave fueron ajustados experimentalmente:

Tasa de aprendizaje (lr)	1e-4
Descuento gamma (γ)	0.99
Tamaño del buffer	10,000 transiciones
Tamaño del batch	32
Frecuencia de actualización del target	cada 10 episodios

4. Código

El código del proyecto se estructuró en módulos independientes para facilitar el mantenimiento y la claridad del flujo. Los siguientes son los principales:

- trading_env.py: define la lógica del entorno personalizado, respetando la interfaz de Gymnasium (reset, step, render, etc.).
- model.py: contiene la definición de la red neuronal con LSTM y la red target.
- train.py: ejecuta el bucle principal de entrenamiento, incluyendo la gestión del buffer y el algoritmo DQN.
- evaluate.py: evalúa el desempeño del agente con datos fuera del conjunto de entrenamiento.
- main.py: orquesta la descarga de datos, el entrenamiento, evaluación y visualización de resultados.

El código completo se encuentra disponible en el githug referenciado en la portada para ser consultado.

5. Resultados obtenidos

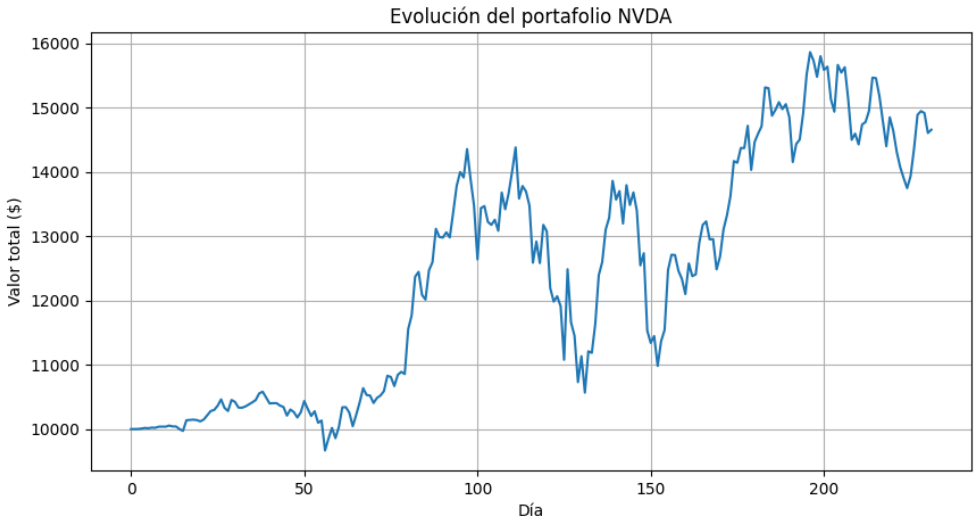
El agente fue entrenado con datos del activo NVDA entre enero de 2020 y diciembre de 2023. Luego se evaluó su rendimiento sobre un período no visto correspondiente al año 2024.

Los resultados obtenidos fueron los siguientes:

- Recompensa total: 4657.90
- Días operados: 231
- Balance final: $85.30
- Acciones restantes: 106
- Valor total final: $14657.90
- Ganancias totales: $30398.09
- Pérdidas totales: $25740.19
- Ganancia / Pérdida neta: $4657.90 (46.58%)
- Profit Factor: 1.18

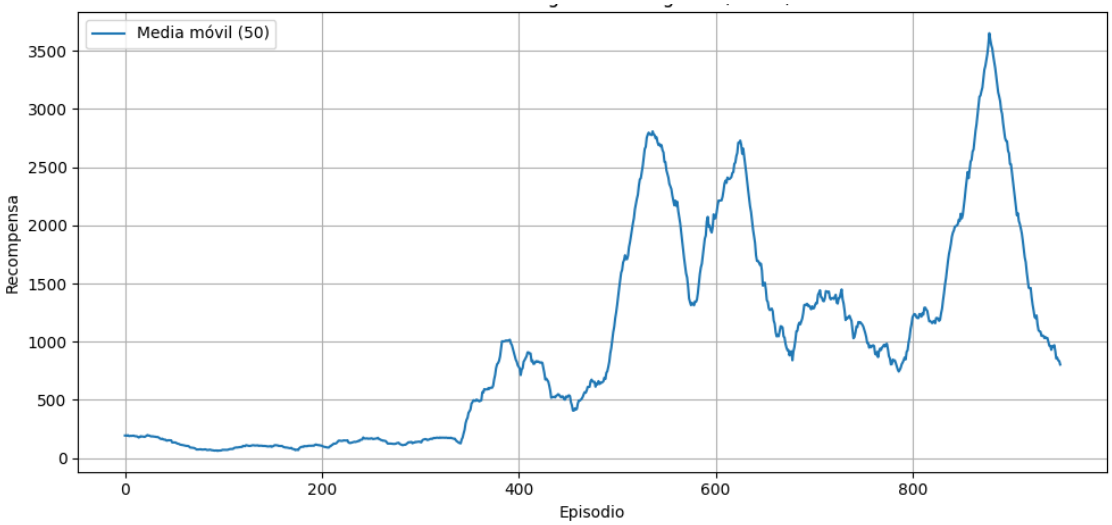
5.1 Evolución del portafolio

Se generó un gráfico que muestra el valor total del portafolio (efectivo + valor de las acciones) a lo largo del período de evaluación. Se observa una **tendencia creciente**, indicativa de que el agente logró una política rentable.



5.2 Evolución del reward

Durante el entrenamiento se monitoreó la recompensa promedio por episodio. Para facilitar su interpretación, se aplicó una **media móvil** que suaviza la curva de convergencia, destacando la mejora sostenida del agente en el tiempo.



6. Conclusiones

Este trabajo logró implementar un agente de trading basado en aprendizaje por refuerzo profundo con resultados satisfactorios en un entorno financiero simulado.

Entre los principales aportes se destacan:

- Diseño de un entorno propio que simula dinámicas del mercado bursátil a partir de datos reales.
- Aplicación de técnicas como experiencia replay, red target y LSTM para mejorar la estabilidad y capacidad de aprendizaje del agente.
- Validación del rendimiento del agente con datos no vistos, evidenciando un comportamiento rentable.

Entre los desafíos más relevantes del proyecto se encontraron:

- La complejidad del código para abordar este tipo de agente que resto tiempo para un mejor ajuste.
- La implementación de un entorno personalizado. Esto implicó diseñar y validar la lógica de las transiciones, recompensas.
- La selección y modelización de una red adecuada para este tipo de problemas.
- El ajuste de hiperparámetros críticos, como la tasa de aprendizaje, el tamaño del buffer y la política de exploración.
- La gestión del balance entre exploración y explotación, fundamental para lograr una política eficiente.

Como trabajo futuro, se propone:

- Explorar arquitecturas más avanzadas, como Transformers o redes híbridas.
- Incorporar indicadores técnicos adicionales y métricas de riesgo (drawdown, Sharpe ratio).
- Extender la simulación a escenarios multi-activo o multi-agente para evaluar estrategias colaborativas o competitivas.