

# Aprendizaje por Refuerzo en Breakout con DQN (Gymnasium + Stable-Baselines3)

## Resumen

Se implementó un agente Deep Q-Network (DQN) para el entorno ALE/Breakout-v5 usando Gymnasium, ale-py y Stable-Baselines3 con CnnPolicy. El agente entrenó durante 20 millones de *timesteps*. La recompensa promedio por episodio pasó de ~1.0 en los primeros miles de pasos a ~35 cerca del final del entrenamiento, acompañado por un aumento en la duración de episodio hacia ~1000 pasos. Las corridas de evaluación muestran episodios con recompensas en el rango 16–45. Se incluyen gráficos de convergencia y distribución de recompensas.



## Introducción

Breakout es un entorno clásico de Atari donde una paleta controla la trayectoria de una pelota para romper ladrillos y maximizar la recompensa. El objetivo es aprender una política que, a partir de observaciones visuales, seleccione acciones discretas (izquierda/derecha/noop/servir) que maximicen el retorno acumulado.

## DQN, Gymnasium y SB3

Se eligió Deep Q-Network (DQN), una extensión de Q-Learning que aproxima la función de acción-valor con una red convolucional, emplea replay buffer para romper correlación temporal y red objetivo para estabilizar las actualizaciones. Esta técnica forma parte del temario propuesto en la consigna.

### Entorno y preprocesado

- Gymnasium + ale-py con el entorno "ALE/Breakout-v5".

- AtariPreprocessing: reescalado a  $84 \times 84$ , escala de grises, *frame\_skip* = 4 (en el *wrapper*, dejando el *env* base en *frameskip*=1), *noop\_max*=30, y *terminal\_on\_life\_loss*=False.
  - FrameStackObservation con *stack\_size*=4 para aportar información temporal.
  - Monitor para registrar métricas en *monitor.csv* (recompensa *r*, longitud *l*, tiempo *t*).
- Este *pipeline* disminuye la dimensionalidad, acelera el aprendizaje y asegura compatibilidad con SB3.

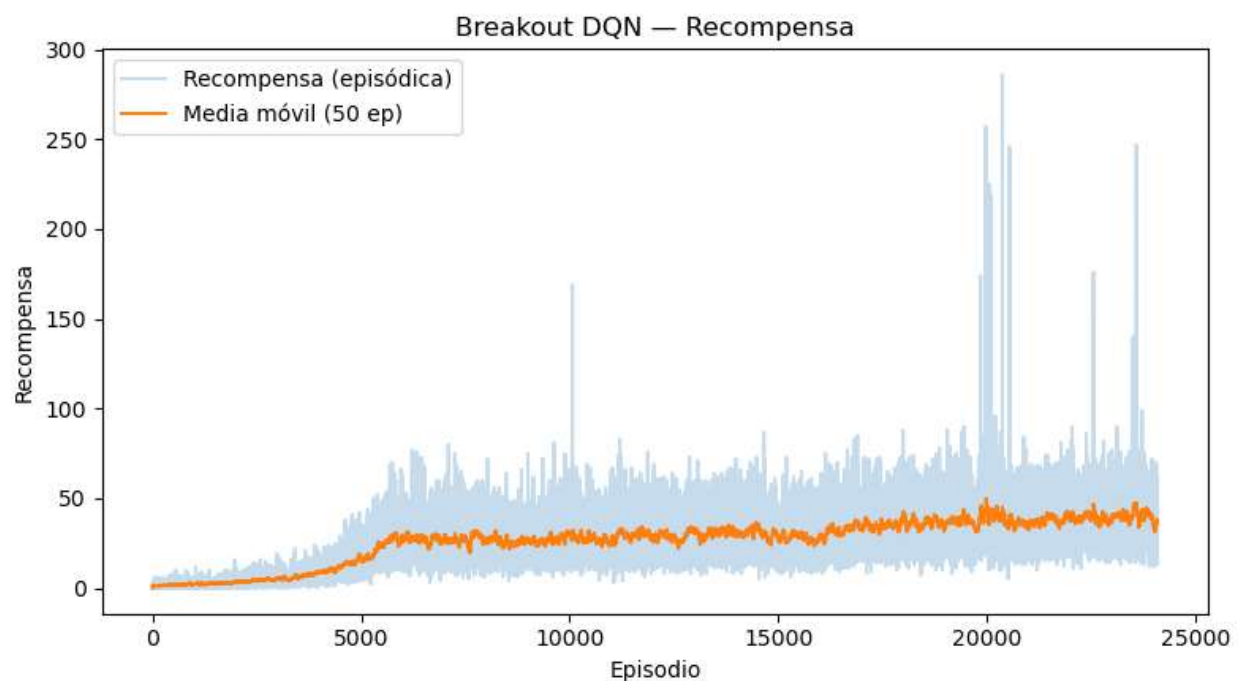
## DQN

- Librería: Stable-Baselines3.
- Política: CnnPolicy estándar de SB3 para Atari.
- Hiperparámetros:

*buffer\_size*=100\_000, *learning\_starts*=10\_000  
*train\_freq*=4, *target\_update\_interval*=1\_000  
*exploration\_fraction*=0.10, *exploration\_final\_eps*=0.01  
*Optimizer* y *learning\_rate*=1e-4

- Presupuesto de entrenamiento: *total\_timesteps*=20\_000\_000.

## Resultados



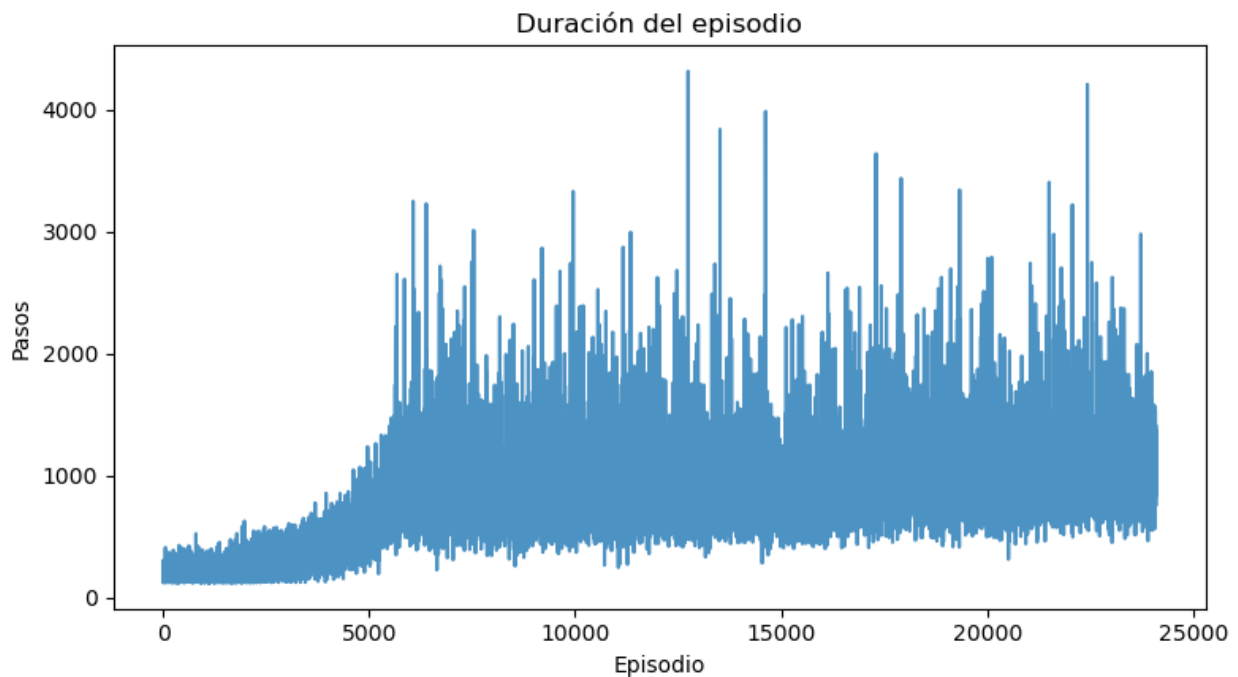
## Entrenamiento (SB3)

Al inicio ( $\leq \sim 12k$  steps):

- $ep\_rew\_mean \approx 1.0-1.6$ ,  $ep\_len\_mean \approx 176-204$ ,  $exploration\_rate \approx 1.0$

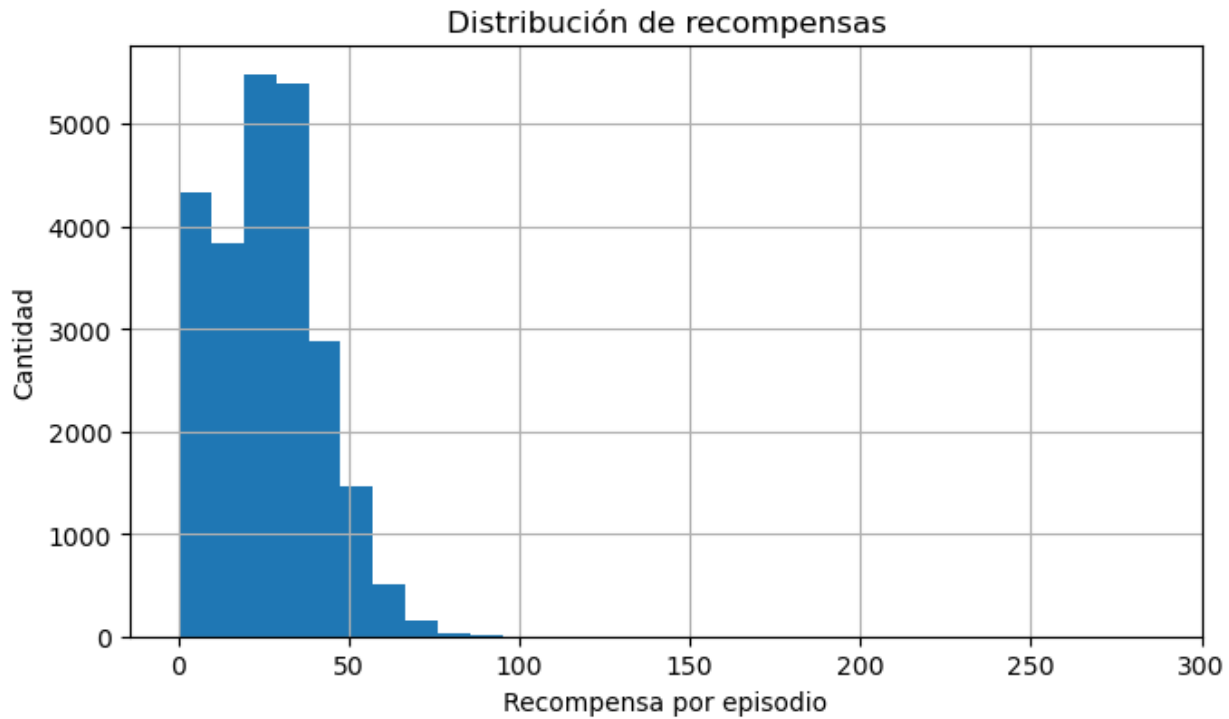
Cerca del final ( $\approx 20M$  steps):

- $ep\_rew\_mean \approx 34.8-35.4$ ,  $ep\_len\_mean \approx 1.00-1.01e3$ ,  $exploration\_rate \approx 0.01$ .
- Actualizaciones totales  $\approx 5.0M$ ,  $loss$  estabilizado  $\sim 7e-3$ .
- Recompensas observadas: 28.0, 45.0, 27.0 y 39.0, 22.0, 16.0 (múltiples corridas).



Interpretación:

- La tendencia creciente de la recompensa y la estabilización del  $loss$  sugieren que el agente aprendió.
- El aumento de longitud de episodio se correlaciona con mayor control de la pelota.



## Conclusiones

El agente DQN entrenado con preprocesado estándar de Atari y una política CNN mejoró su desempeño, alcanzando una recompensa promedio por episodio  $\sim 35$  tras 20M *timesteps*. La evidencia en las trazas y figuras indica convergencia hacia una política estable que prolonga los episodios y acumula mayor retorno. Para un trabajo de curso, los resultados son sólidos y reproducibles con la configuración presentada.