Guía Práctica de Escalado de Parámetros PPO para Entrenamiento

13 de octubre de 2025

1. Introducción

Esta guía explica cómo ajustar los principales parámetros de entrenamiento del algoritmo **PPO (Proximal Policy Optimization)** al aumentar el tamaño del grafo en un entorno de navegación por waypoints, cuando el entrenamiento se realiza con **una única GPU**. El objetivo es mantener la estabilidad del aprendizaje y aprovechar al máximo los recursos disponibles, evitando saturar memoria y tiempo de cómputo innecesariamente.

2. Tabla de Referencia Rápida

Cuadro 1: Valores iniciales recomendados por tamaño de grafo

Tamaño Grafo	Nodos	max_steps	total_timesteps	eval_freq	lr	ent_coef
5×5	25	30	2×10^5	5×10^3	3×10^{-4}	0.05
10×10	100	80	5×10^5	10×10^3	3×10^{-4}	0.05
20×20	400	200	1×10^6	25×10^3	1×10^{-4}	0.07
25×25	625	250	$1{,}5 \times 10^6$	25×10^3	1×10^{-4}	0.07
50×50	2,500	600	3×10^6	50×10^3	5×10^{-5}	0.10

3. Parámetros Clave y Escalado Práctico

3.1. 1. Límite de pasos por episodio (`max_steps`)

Controla cuántas acciones puede ejecutar el agente antes de reiniciar un episodio. Debe crecer con el tamaño del grafo, ya que hay más nodos que recorrer.

$$\texttt{max\_steps} \approx N_{\text{nodos}} \times (0{,}3 \text{ a } 0{,}4)$$

Ejemplos:

- 5×5: $25 \times 1{,}2 = 30$
- 20×20: $400 \times 0{,}5 = 200$
- 50×50: $2500 \times 0{,}24 = 600$

Consejo práctico: Si el agente se reinicia antes de alcanzar los waypoints, aumenta `max_steps`; si se estanca o repite ciclos, redúcelo levemente.

3.2. 2. Duración total del entrenamiento (`total_timesteps`)

Cantidad total de pasos de interacción entre el agente y el entorno durante todo el entrenamiento. Escala directamente con el número de nodos:

$$\texttt{total\_timesteps} \approx N_{\text{nodos}} \times (1000 \text{ a } 2000)$$

Ejemplos:

- 5×5: $25 \times 8,000 = 200,000$
- 20×20: $400 \times 2,500 = 1,000,000$
- 50×50: $2500 \times 1,200 = 3,000,000$

Consejo: Si el entrenamiento converge rápido y la recompensa se estabiliza, no es necesario agotar todos los timesteps.

3.3. 3. Frecuencia de evaluación (`eval_freq`)

Número de pasos entre cada evaluación del modelo. Debe elegirse de forma que haya alrededor de 40 evaluaciones por entrenamiento:

$$\texttt{eval\_freq} \approx \frac{\texttt{total\_timesteps}}{40}$$

Ejemplos:

- 5×5: $200,000/40 = 5,000$
- 20×20: $1,000,000/40 = 25,000$
- 50×50: $3,000,000/40 = 75,000$

Práctico: con una sola GPU, usar evaluaciones menos frecuentes (cada 50k) evita pausas excesivas.

3.4. 4. Tasa de aprendizaje (`learning_rate`)

Controla la magnitud de las actualizaciones del modelo. Valores altos aceleran el aprendizaje pero pueden volverlo inestable.

$$\text{Grafos pequeños } (< 100) : 3 \times 10^{-4}$$

$$\text{Grafos medianos } (100 - 500) : 1 \times 10^{-4} \text{ a } 3 \times 10^{-4}$$

$$\text{Grafos grandes } (> 500) : 5 \times 10^{-5} \text{ a } 1 \times 10^{-4}$$

Recomendación: comenzar con 3×10^{-4} y reducir gradualmente si las curvas de recompensa oscilan demasiado.

3.5. 5. Coeficiente de entropía (`ent_coef`)

Aumenta la exploración penalizando políticas muy deterministas. Crucial cuando el grafo tiene caminos alternativos o regiones similares.

$$\text{Pequeños: } 0{,}05 \qquad \text{Medianos: } 0{,}05 - 0{,}07 \qquad \text{Grandes: } 0{,}07 - 0{,}10$$

Consejo: si el agente se queda en ciclos o repite trayectorias, subir `ent_coef`; si explora demasiado y no converge, bajarlo.

3.6. 2.5. Análisis de la Política Determinista y Entropía

Una **política muy determinista** en el Aprendizaje por Refuerzo (RL) es aquella que, dado un estado específico en el grafo, **casi siempre elige la misma acción**. En otras palabras, la política tiene muy poca o ninguna incertidumbre. Se opone a una política estocástica (o probabilística), donde el agente podría elegir diferentes acciones para el mismo estado con cierta probabilidad.

Naturaleza y Riesgos

¿Qué la Hace Determinista? El agente de PPO utiliza una red neuronal para estimar la probabilidad de cada acción dado un estado. Cuando una política es muy determinista, significa que la *Probabilidad de la acción elegida* ≈ 1 y la *Probabilidad de las otras acciones* ≈ 0. Esto se debe a que el modelo ha reducido la **entropía** de la distribución de probabilidad de las acciones a un valor muy bajo.

El Riesgo en Grafos Grandes (Ejemplo 100×100**):** En entornos complejos, una política muy determinista presenta un riesgo significativo:

- **Óptimos Locales:** El agente encuentra una ruta funcional (un óptimo local) y se "pega" a ella, ignorando cualquier otra ruta que podría ser significativamente mejor (el óptimo global).
- **Falta de Exploración:** Si la política siempre elige el mismo camino, nunca visita estados nuevos o poco explorados, impidiendo descubrir soluciones más cortas o eficientes.

- **Comportamiento Rígido:** La política puede volverse vulnerable a pequeños cambios en el entorno (ej., un obstáculo temporal), ya que no tiene la flexibilidad (estocasticidad) para adaptarse.

Relación con el Coeficiente de Entropía (`ent_coef`)

El `ent_coef` añade un término a la función de pérdida de PPO que **penaliza al agente si su política se vuelve demasiado determinista**.

- **Aumentar `ent_coef` (ej., a 0.10):** Se recompensa al agente por mantener una distribución de acciones más uniforme (más estocástica). Esto fuerza al agente a **explorar activamente** en el vasto grafo y es crucial para evitar caer en óptimos locales.
- **Reducir `ent_coef` (ej., a 0.01):** Se permite que el agente se vuelva muy determinista. Esto es útil cerca del final del entrenamiento, cuando se asume que se encontró la mejor solución y solo se necesita **explotar** ese conocimiento con una política de alta confianza.

3.7. 6. Factores de descuento (`gamma` y `gae_lambda`)

Valores recomendados para la mayoría de los entornos:

$$\texttt{gamma} = 0{,}99 \quad \text{y} \quad \texttt{gae\_lambda} = 0{,}95$$

Interpretación: equilibrio entre estabilidad y capacidad de aprendizaje a largo plazo. No requieren ajuste fino salvo casos extremos.

3.8. 7. Criterio de parada temprana (`max_no_improvement_evals`)

Número máximo de evaluaciones consecutivas sin mejora antes de detener el entrenamiento.

$$\texttt{max\_no\_improvement\_evals} = 10 \text{ a } 15$$

Consejo práctico: con una sola GPU, esto evita malgastar horas de cómputo en convergencias falsas.

4. Indicadores para Reajustar Parámetros

4.1. Problemas comunes

- **El agente no aprende:** Recompensa negativa constante. → Aumentar `max_steps` o `ent_coef`.
- **Aprende muy lento:** → Aumentar `total_timesteps` o `learning_rate`.

- **Entrenamiento inestable:** → Reducir `learning_rate` o `clip_range`.
- **Se queda en ciclos:** → Aumentar `ent_coef` o reducir `max_steps`.

4.2. Señales de buen entrenamiento

- Recompensa promedio aumenta consistentemente.
- Early stopping se activa entre 50k y 100k pasos.
- El agente completa los waypoints en menos del 30 % de los pasos máximos.

5. Checklist de Escalado

1. Definir el nuevo tamaño del grafo ($N \times N$).
2. Calcular `max_steps` según el número de nodos.
3. Ajustar `total_timesteps` y `eval_freq`.
4. Revisar `learning_rate` y `ent_coef`.
5. Configurar early stopping con 10–15 evaluaciones sin mejora.
6. Ejecutar una corrida corta de prueba antes del entrenamiento completo.

6. Conclusión

Con una sola GPU, el éxito del entrenamiento depende más del **ajuste gradual de parámetros** y de la **monitorización constante** que de la potencia bruta. El escalado correcto de `max_steps`, `total_timesteps` y `ent_coef` permite mantener la estabilidad del aprendizaje incluso en grafos grandes, garantizando eficiencia y reproducibilidad sin necesidad de múltiples GPUs.