UNIVERSIDAD DEL VALLE DE GUATEMALA
Deep Learning
Sección 10

Proyecto Final Deep Learning

Diego Valdez 21328
Joaquín Puente 22296
Nelson García 22434

Guatemala, 10 de noviembre de 2025

Descripción del problema

Este proyecto implementa un entorno de carreras basado en grids, lo que se busca es que un vehículo autónomo debe de navegar por una pista cargada desde archivos CSV. La tecnología utilizada fue el aprendizaje por refuerzo utilizando DQN. El entorno simula una carrera donde el carro avanza horizontalmente hacia la derecha, este tiene control sobre la dirección lateral y la aceleración. La pista como tal tiene 6 tipos de objetos diferentes que serían pavimento, muros con los que puede chocar, áreas exteriores, aceite que reduce velocidad un 10%, terracería un 5% y boost que aumenta temporalmente la velocidad en 5%. El sistema de recompensas está diseñado para incentivar el progreso hacia la derecha, mantener el vehículo cerca del centro del carril, minimizar el tiempo de carrera, y penalizar fuertemente las colisiones con muros, ofreciendo un gran bonus al alcanzar la meta ubicada cerca del borde derecho de la pista.

Análisis

El problema principal que enfrenta el agente es aprender una política de navegación que balancee velocidad, seguridad y precisión en un entorno parcialmente observable con consecuencias terminales por colisión. El agente debe aprender a anticipar obstáculos y planificar maniobras con información local, similar a conducir con niebla. El sistema de recompensas presenta un balance que funciona de la siguiente manera, la penalización constante por tiempo presiona al agente a ir rápido, pero velocidades altas reducen el margen de maniobra ante obstáculos, mientras que la penalización por desviación del centro compite con la necesidad de esquivar muros y aceite. Las superficies especiales añaden complejidad adicional ya que sus efectos sobre la velocidad son inmediatos y pueden alterar drásticamente el comportamiento esperado del vehículo, requiriendo que el agente aprenda modelos internos de estas dinámicas.

Propuesta de solución

Se propone mejorar el sistema mediante modificaciones en tres áreas clave. Primero, se debe mejorar el espacio de observación añadiendo dos canales adicionales al patch: uno que represente la velocidad normalizada del vehículo y otro que indique el offset lateral respecto al centro del carril. Estos canales proporcionarán al agente información del estado que facilita el aprendizaje de políticas de control más efectivas. Segundo, se recomienda ajustar la función de recompensas para balancear mejor los objetivos: incrementar el coeficiente de progreso para enfatizar el avance rápido, modificar la penalización por desviación para que sea más tolerante y reducir ligeramente la penalización por tiempo para permitir maniobras defensivas cuando sea necesario. Tercero, se debe implementar un límite máximo de pasos por episodio (truncated=True después de, por ejemplo, 500 pasos) para prevenir episodios infinitos y forzar al agente a desarrollar estrategias eficientes. Cuarto, se sugiere entrenar el agente gradualmente comenzando con pistas simples antes de introducir pistas más complejas, permitiendo que el agente construya habilidades básicas de control antes de enfrentar desafíos avanzados.

Descripción de la solución

La solución implementada consiste en un sistema de aprendizaje por refuerzo donde un agente autónomo aprende a navegar por pistas de carreras mediante interacción directa con el entorno. El enfoque adoptado utiliza observaciones que capturan el contexto local alrededor del vehículo en cada momento, codificando la información visual de la pista en formato matricial con múltiples canales que representan los diferentes tipos de superficie. El agente toma decisiones discretas sobre dirección y aceleración basándose en estas observaciones, recibiendo retroalimentación en forma de recompensas que reflejan qué tan bien está cumpliendo los objetivos de la tarea: avanzar rápidamente hacia la meta, mantenerse cerca del centro del carril, y evitar colisiones. El sistema de recompensas combina múltiples señales incluyendo progreso incremental por cada unidad avanzada, penalizaciones proporcionales al tiempo transcurrido y a la desviación lateral, más recompensas terminales significativas al alcanzar la meta o penalizaciones severas por chocar. La dinámica del vehículo simula efectos realistas de diferentes superficies sobre la velocidad, donde aceite y terracería reducen el rendimiento mientras que tiles especiales de boost proporcionan aceleración temporal. El proceso de entrenamiento permite al agente explorar diferentes estrategias inicialmente y gradualmente converger hacia políticas más determinísticas y efectivas conforme acumula experiencia, utilizando episodios completos desde el inicio hasta colisión o meta para actualizar su conocimiento sobre qué acciones son beneficiosas en diferentes situaciones.

Herramientas aplicadas

Deep Q-Network: Se utilizó el algoritmo DQN como método principal de aprendizaje por refuerzo, el cual aproxima la función Q óptima mediante una red neuronal profunda. Este algoritmo es particularmente apropiado para el problema porque maneja eficientemente espacios de acciones discretos y puede aprender políticas complejas en entornos con alta dimensionalidad de estados.

Red Neuronal Convolucional: Se implementó una arquitectura CNN personalizada como extractor de características visuales, compuesta por múltiples capas convolucionales con funciones de activación ReLU que procesan las observaciones egocéntricas en formato imagen. La red utiliza kernels de 3×3 con diferentes configuraciones de stride y padding para capturar patrones espaciales a múltiples escalas, seguidas por pooling adaptativo que reduce las dimensiones espaciales a un vector de características fijo.

Gymnasium: Se empleó el framework Gymnasium como interfaz estándar para definir el entorno de carreras, proporcionando una API consistente con métodos reset() y step() que facilitan la interacción entre agente y entorno. Esta herramienta estandariza la definición de espacios de observación y acción, maneja automáticamente aspectos como terminación de episodios y truncamiento, y permite integración directa con algoritmos de Stable-Baselines3.

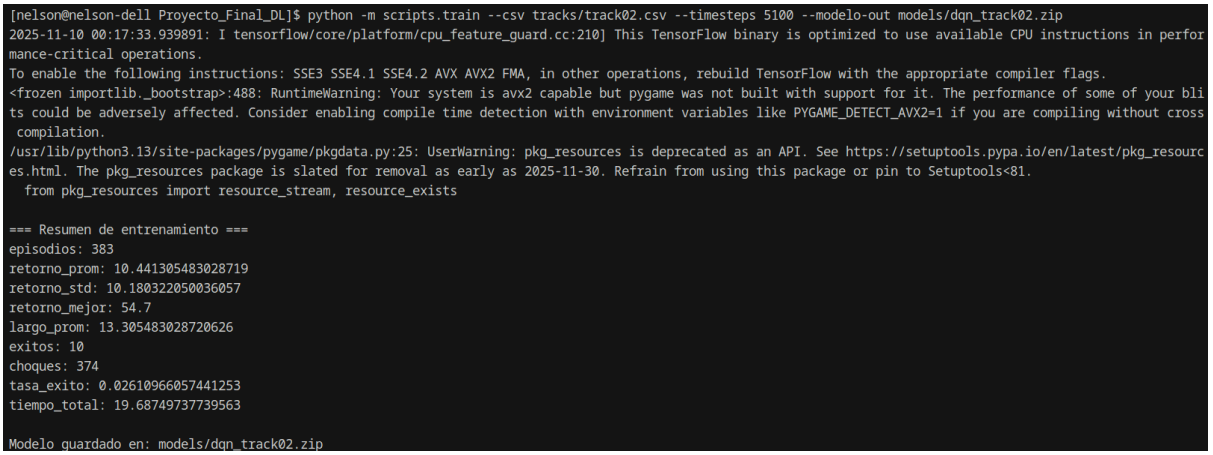
Stable-Baselines3: Se utilizó esta biblioteca de alto nivel que proporciona implementaciones optimizadas y probadas de algoritmos de aprendizaje por refuerzo, incluyendo DQN. La biblioteca maneja automáticamente aspectos complejos como la gestión del replay buffer,

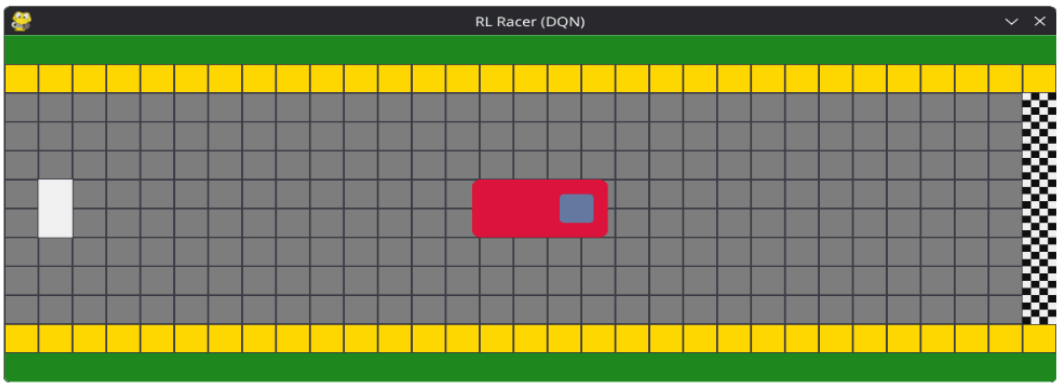
actualización de redes objetivo, decaimiento de epsilon para exploración, y optimización mediante gradiente descendente, permitiendo enfocarse en el diseño del entorno y ajuste de hiperparámetros en lugar de implementar estos mecanismos desde cero.

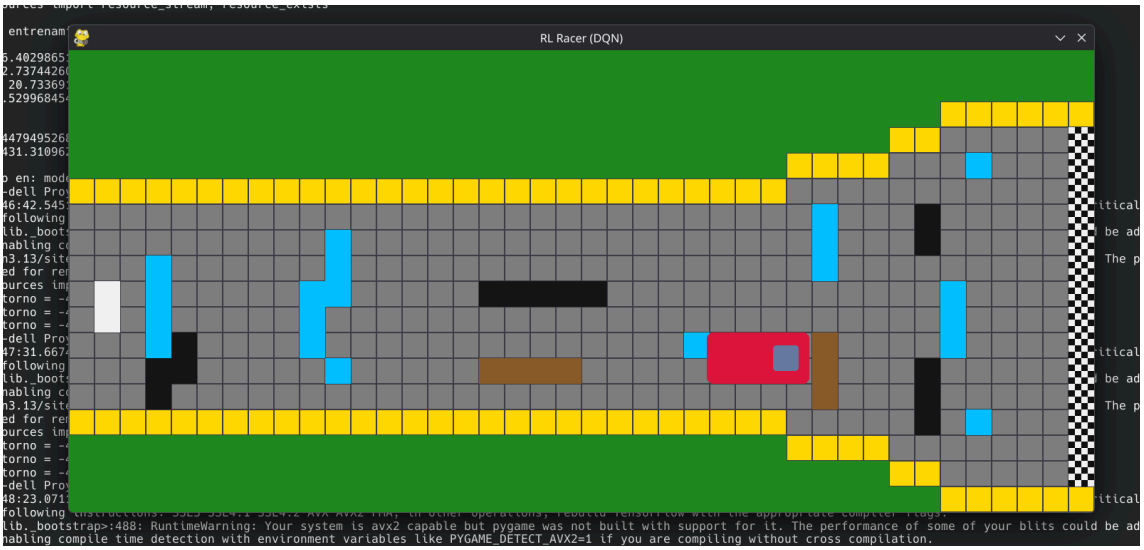
PyTorch: Sirvió como backend para todas las operaciones de deep learning, manejando la construcción de redes neuronales, cálculo de gradientes mediante backpropagation, y optimización de parámetros.

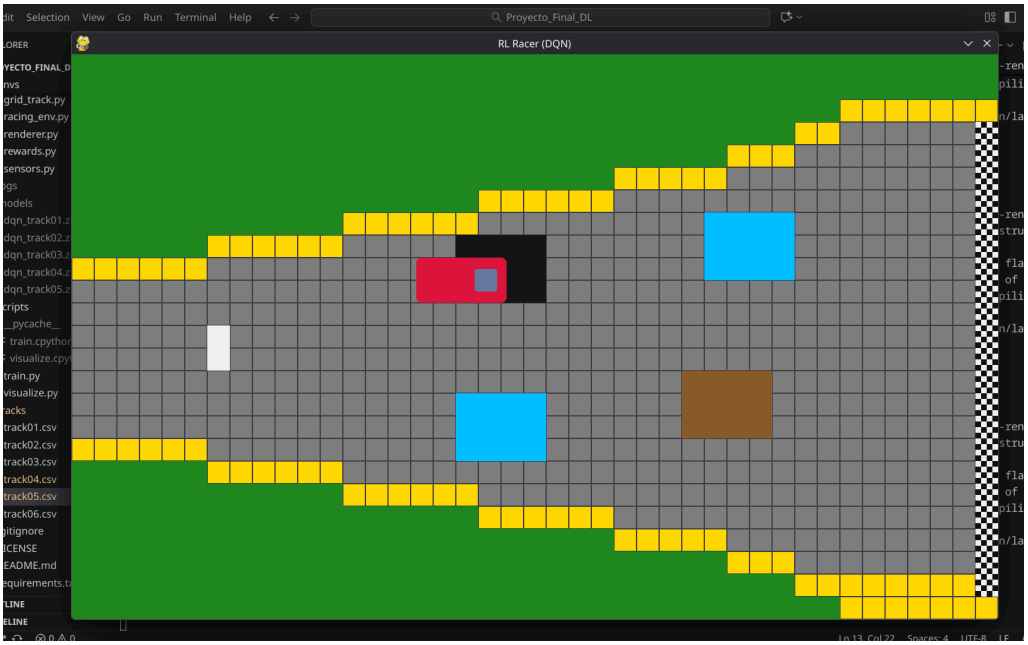
Pygame: Se incorporó para visualización en tiempo real del progreso del agente durante entrenamiento y evaluación, renderizando la pista, posición del vehículo, y trayectoria seguida.

Pruebas:









Resultados

Track	Episodios	Tasa de éxito	Tiempo Total (s)
1	**214**	**45.76%**	**4.8**
2	**294**	**55.78%**	**67**
3	**821**	**57.91%**	**2812**
4	**1919**	**59.67%**	**3069**

Track	Episodios	Tasa de éxito	Tiempo Total (s)
1	**5345**	**55.21%**	**4392**
2	**5987**	**58.43%**	**4459**
3	**5363**	**60.77%**	**4731**
4	**6423**	**57.92%**	**4932**

Conclusión

- La implementación de Deep Q-Network demostró ser efectiva para el problema de navegación autónoma en pistas de carreras con acciones discretas, logrando aprender políticas que balancean velocidad, seguridad y precisión de trayectoria mediante la aproximación de funciones Q con redes neuronales convolucionales que procesan observaciones egocéntricas del entorno.
- El diseño del sistema de recompensas resultó fundamental para guiar el comportamiento del agente, combinando progreso incremental, penalizaciones por tiempo y desviación lateral, junto con recompensas terminales.
- La arquitectura modular del código con separación clara entre componentes facilitó el desarrollo iterativo y permite extensiones futuras, demostrando que las buenas prácticas de ingeniería de software aplicadas a proyectos de aprendizaje por refuerzo mejoran significativamente la mantenibilidad y escalabilidad del sistema.
- La simulación de efectos de superficie como aceite, terracería y boost añade realismo y complejidad al entorno, requiriendo que el agente aprenda modelos internos de estas dinámicas para anticipar cambios en el comportamiento del vehículo, lo cual demuestra la capacidad del DQN para capturar relaciones causa y efecto complejas a partir de experiencia directa.

Bibliografía

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533. https://doi.org/10.1038/nature14236

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep Reinforcement Learning with Double Q-learning. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1), 2094-2100. https://doi.org/10.1609/aaai.v30i1.10295

Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press. http://incompleteideas.net/book/the-book-2nd.html

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., & Dormann, N. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 22(268), 1-8. https://jmlr.org/papers/v22/20-1364.html

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym. arXiv preprint arXiv:1606.01540. https://arxiv.org/abs/1606.01540