Laboratorio RL-03: Implementación y pruebas de VizDoom

Sebastian Moscoso Riveros
Departamento de Ciencia de la Computación
Universidad Católica San Pablo
sebastian.moscoso@ucsp.edu.pe

Junio 2022

1. Introducción

El siguiente informe se detalla la Implementación y pruebas de VizDoom, para poder enteder mejor se realizaron algunos experimentops en dos juegos y se contestaron algunas preguntas.

2. Preguntas

2.1. ¿Cómo es el modelo de mundo utilizado: MDP, Modelo de premios y modelo Q-Learning ?

Como sabemos un modelo MDP esta formado por el conjunto de estados, acciones , modelo de transición y premios.
El proyecto de VizDoom nos presenta dos experimentos en donde cada uno presenta un modelo MDP :

2.1.1. Cacodemon

- Estados: Dado por episodios que duran 300 frames osea 300 estaods, o puede ser el estado finalizador que es el asesinar al monstruo.
- Acciones: Ir a la derecha, Ir a la izquierda, disparar
- Modelo de transición: No tiene porque es model-free

- Premios: El agente obtiene 101 puntos por matar al monstruo, -5 por fallar el tiro, y, además, -1 por cada acción

2.1.2. Medikit Collecting

- Estados: Cada episodio termina después de 2100 ticks (1 minuto en tiempo real) o cuando el agente muere, por lo que 2100 es la puntuación máxima que se puede alcanzar.
- Acciones: El agente puede moverse (adelante/atrás), y girar (izquierda/derecha), 4 acciones.
- Modelo de transición: No tiene porque es model-free
- Premios: Se anota 1 punto por cada tic, y se castiga con -100 puntos por morir. Recompensas en forma de 100 y -100 puntos por recoger un botiquín y una ampolla, respectivamente

Ademas ambos experimentos presenta un modelo q-learning para aprender la politica de las acciones. La acción es seleccionada por una política -greedy con decaimiento lineal. La función Q se aproxima con una red neuronal convolucional, que se entrena con Stochastic Gradient Decent.

2.2. ¿Qué tipo de política es usada?

La pólitica usada es e-greedy con decaimiento lineal. La política Epsilon-greedy, con probabilidad epsilon elige una acción aleatoria y la mejor acción (al menos según las estimaciones actuales) con probabilidad 1-epsilon. Con el fin de que el agente realizará acciones completamente aleatorias (exploración) y comenzará a hacer algo razonable más adelante (explotación).

2.3. ¿Cómo se calculan y aproximan los valores Q?

La función Q se aproxima con una red neuronal convolucional, que se entrena con Stochastic Gradient Decent.
La red utilizada en el experimento Cacodemon consta de dos capas convolucionales con 32 filtros cuadrados de 7 y 4 píxeles de ancho, respectivamente. A cada capa convolucional le sigue una capa de agrupación máxima de tamaño 2 y unidades lineales rectificadas para la activación. A continuación, hay una capa totalmente conectada con 800 unidades lineales rectificadas con fugas y una capa de salida con 8 unidades lineales correspondientes a

las 8 combinaciones de las 3 acciones disponibles acciones disponibles (izquierda, derecha y disparo).
La red utilizada en el experimento Medikit Collecting se trata de tres capas convoluciones con 32 filtros cuadrados de 7, 5 y 3 píxeles de ancho, respectivamente. La capa totalmente conectada utiliza 1024 unidades lineales rectificadas con fugas y la capa de salida 16 unidades lineales correspondientes a cada combinación de las 4 acciones disponibles.

2.4. ¿ Qué modelo de optimización se usa?

El modelo de optimización que usa VizDoom es Deep Q-learning El problema se modela como un Proceso de Decisión de Markov y se utiliza Qlearning para aprender la política.

2.5. ¿Existe estrategia de repetición de experiencias?

VizDoom usa la la repetición de la experiencia en sus dos experimentos con una capacidad de memoria de repetición de hasta 10 000 elementos.

2.6. ¿Existen objetivos Q fijos?

Si existen objetivos fijos porque al llegar a ese objetivo se gana el juego. En el experimento de Cacodemon se logra matando al monstruo, y el experimento Medikit Collecting se logra obteniendo 2100 puntos.

3. Experimentos

Cada modelo se entreno en un maquina local de sistema operativo linux ubuntu version 20.0. Las caracteristicas del computador son:

- Un procesador AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx
- 16 GB de memoria RAM

3.1. Cacodemon

Para entrenar nuestro agente en el juego de Cacodemon se uso los siguientes parametros:

- aprendizaje de 0.00025

- factor de descuento de 0.99
- épocas 20
- tamaño de repetición de experiencia es de 10000

El tiempo de ejecución del entrenamiento fue de 104.87 minutos como se puede ver en la imagen 1 en cuanto el uso de memoria y cpu no fue mucha porque se ejecuto en un proceso aparte y no hubo problemas para su ejecución.

Figura 1: Entrenamiento de la ultima época

La convergencia se dio después de 20 épocas en donde en la primera época nos encontramos con un aprendizaje muy malo porque se obtiene -145 de pontaje, pero a partir de la 5 época se van dando resultados positivos, es así que en la época 20 se llega a tener un promedio de 80, lo que es promedio elevado para ganar el juego, en la imagen 2 se puede apreciar los resultados de la pruebas realizadas del entrenamiento.

Figura 2: Pruebas del modelo entrenado

3.2. Medikit Collecting

Para entrenar nuestro agente en el juego de Medikit Collecting se uso los siguientes parametros:

- aprendizaje de 0.00025
- factor de descuento de 0.99
- épocas 20
- tamaño de repetición de experiencia es de 10000

El tiempo de ejecución del entrenamiento fue de 214.26 minutos como se puede ver en la imagen 3 en cuanto el uso de memoria y cpu no fue mucha porque se ejecuto en un proceso aparte y no hubo problemas para su ejecución.

Figura 3: Entrenamiento de la ultima época

La convergencia se dio después de 20 épocas en donde en la primera época nos encontramos con un aprendizaje muy malo porque se obtiene 305 de puntaje, pero a partir de la décima época se van dando resultados promedios positivos, es así que en la época 20 se llega a tener un promedio de 1608, lo que es promedio elevado para ganar el juego, en la imagen 4 se puede apreciar los resultados de la pruebas realizadas del entrenamiento.

4. Link del github

Link del github: https://github.com/semr9/vizdoom-project.git.

Referencias

[1] M. Wydmuch, M. Kempka, and W. Jaśkowski, "Vizdoom competitions: Playing doom from pixels," *IEEE Transactions on Games*, 2018.

Figura 4: Pruebas del modelo entrenado