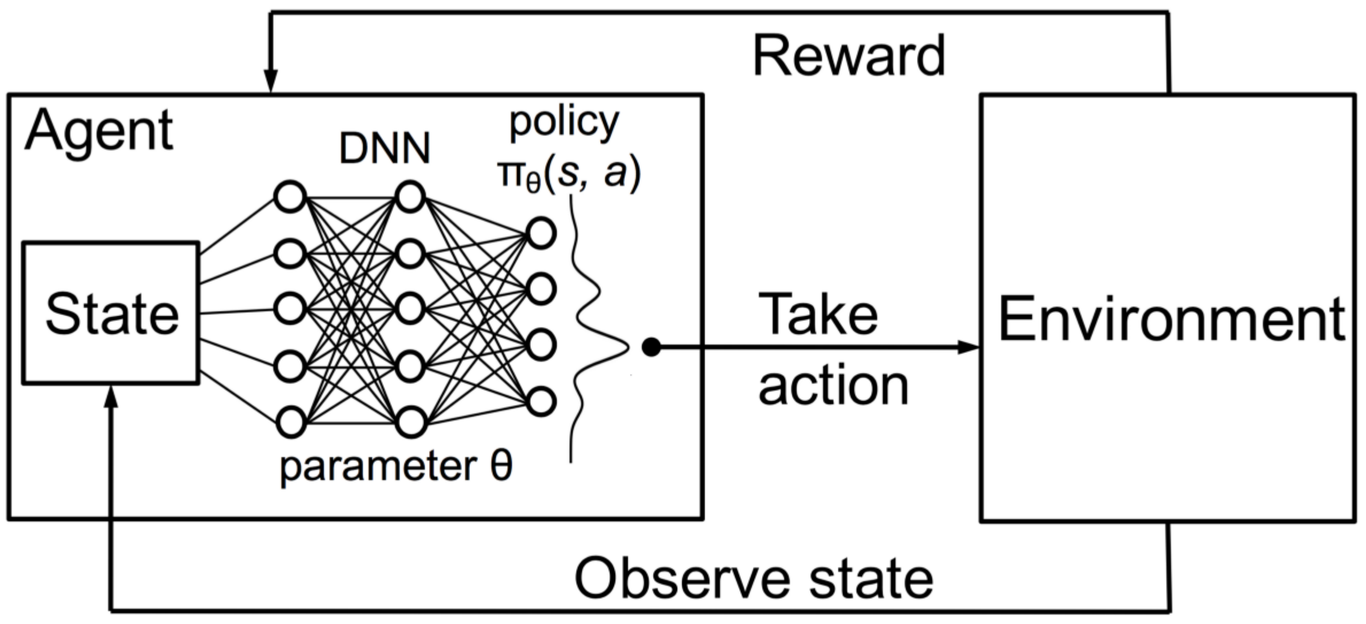
PONG Deep Q Learning

Martín Sebastián Rodríguez Turco

PONG Deep Q Learning se basa en la implementación de redes neuronales junto con un sistema de recompensas para que un agente que desconoce por completo su ambiente aprenda las estrategias que más lo benefician.



Introducción

El proyecto se desarrolló con el fin de mostrar las capacidades de las redes neuronales para enfrentar desafíos y aprender de su entorno. Este tipo de técnicas permite reducir la dificultad de programar un agente para que realice tareas complejas.

Objetivo

Entrenar un agente autónomo para que pueda ser competente en el juego de Pong comprendiendo su funcionamiento sin tener conocimiento previo acerca del juego.

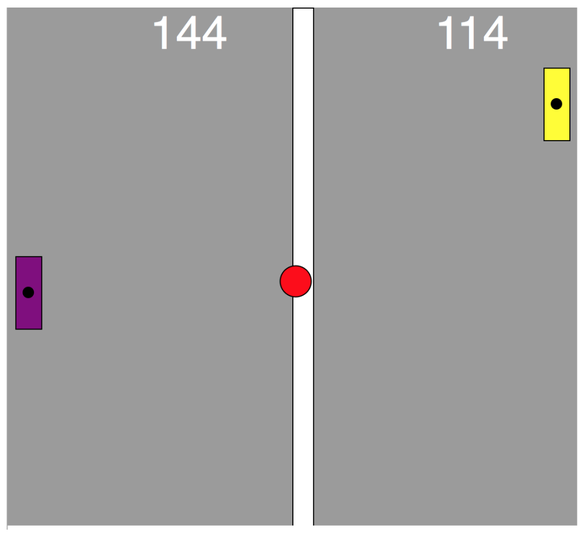
Métodos

El proyecto comenzó con la implementación del algoritmo de Deep Q Learning y la realización del entorno de simulación.

Deep Q Learning permite entrenar una red neuronal mediante la observación del entorno y los estímulos que el agente recibe de él (recompensas o castigos). El agente comienza en la etapa de **exploración.** En esta etapa el agente descubre por primera vez qué hacen los controles del juego e intenta pasar por la máxima cantidad de estados posibles para estimar cuales son las acciones a tomar dado su estado actual.

La segunda etapa se conoce como **explotación.** Esta se caracteriza por el uso de lo aprendido, es decir haciendo estimaciones de cuál es la próxima acción que maximiza la recompensa esperada.

Tanto en la etapa de exploración como en la de explotación el agente realiza el entrenamiento de la red neuronal durante el juego a a partir de sus estados. La información es ingresada al sistema en tiempo real a comparación de otros métodos den aprendizaje en los cuales es necesario primero recolectar los datos.



Captura del juego en progreso. Izquierda robot de entrenamiento, derecha Agente

La topología de la red fue elegida en función de la dimensionalidad del problema. Este sistema se alimenta de 4 variables posicionales y estima la recompensa de tomar cada una de las 3 acciones disponibles para el agente (*arriba*, *quieto*, *abajo*). Por lo que se decidió implementar una red neuronal profunda de 3 capas y baja cantidad de nodos.

Una vez estimada la recompensa por cada potencial acción a tomar, el agente decidirá llevar a cabo aquella que maximice su recompensa.

Para evitar el overfiting (memorización) y permitir que el agente continúe explorando el entorno aún después de la etapa de exploración. Se hace uso de un coeficiente de decisión aleatoria. Este juega el papel de hacer que el agente tome decisiones aleatorias e ignore por completo sus predicciones. Este coeficiente es sustancialmente importante para el entrenamiento.

Resultados

El agente diseñado presenta problemas de consistencia debido a factores aleatorios en la inicialización de la red. Sin embargo ha tenido muy buenos resultados cuando la inicialización predisponía el aprendizaje.

Conclusiones

Se logró diseñar un agente capaz de aprender de su entorno mediante el uso de redes neurales combinadas con un sistema de recompensas. Estos sistemas se despeñan notablemente en entornos demasiado complejos como para modelarse de manera tradicional.

Referencias / Bibliografía

[1] J. Momoh, *Smart Grid: Fundamentals of Design and Analysis*. NJ, Wiley, 2012.

[2] Santos, L. (2018). *Deep Q Learning · Artificial Inteligence*. [online] Leonardoaraujosantos.gitbooks.io. Available at: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_q_learning.html [Accessed 13 Aug. 2018].

[3] Simonini, T. (2018). *An introduction to Deep Q-Learning: let's play Doom*. [online] freeCodeCamp. Available at: https://medium.freecodecamp.org/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8 [Accessed 15 Aug. 2018].