Nombre: Ricardo Jara

Fecha: 23/05/202

Materia: Sistemas Expertos

Tema: OpenIA

~~~~~~

OpenIA

OpenIA pondrá la tecnología IA en beneficio de la humanidad e intentará evitar ese miedo que existen ante los peligros de la inteligencia artificial y sus malos usos. OpenIA se limitará a aplicarla para facilitarnos la vida a todos, Siendo una plataforma de software para medir y entrenar la inteligencia general de una IA a través del suministro mundial de juegos, sitios web y otras aplicaciones, permite a entrenar y evaluar agentes de IA en una gama extremadamente amplia de entornos complejos en tiempo real.

OpenIA permite que cualquier programa existente se convierta en un entorno OpenAI Gym, sin necesidad de acceso especial a los componentes internos, el código fuente o las API del programa. Lo hace empacando el programa en un contenedor Docker y presentando a la IA con la misma interfaz que utiliza un humano: enviando eventos de teclado y mouse, y recibiendo píxeles de pantalla. Nuestra versión inicial contiene más de 1,000 entornos en los que un agente de IA puede tomar medidas y recopilar observaciones, permite también, algunos entornos incluyen una señal de recompensa enviada al agente, para guiar el aprendizaje de refuerzo. Hemos incluido unos cientos de entornos con señales de recompensa.

GYM: Tiene tres categorías de aprendizaje: son supervisadas, no supervisadas y de refuerzo. En el aprendizaje supervisado:

- Intentamos predecir un valor objetivo o una clase donde los datos de entrada para el entrenamiento ya tienen etiquetas asignadas.
- Mientras que el aprendizaje no supervisado utiliza datos no etiquetados para observar patrones para hacer clusters, PCA o detección de anomalías.
- Los algoritmos RL son procedimientos de optimización para encontrar los mejores métodos para obtener la máxima recompensa, es decir, dar una estrategia ganadora para alcanzar el objetivo, por refuerzo, un agente actúa en el entorno dado de manera continua o discreta para maximizar alguna noción de recompensa codificada en él. Suena demasiado profundo, bueno, es con una base de investigación que se remonta a la psicología conductista clásica, la teoría de juegos, los algoritmos de optimización.

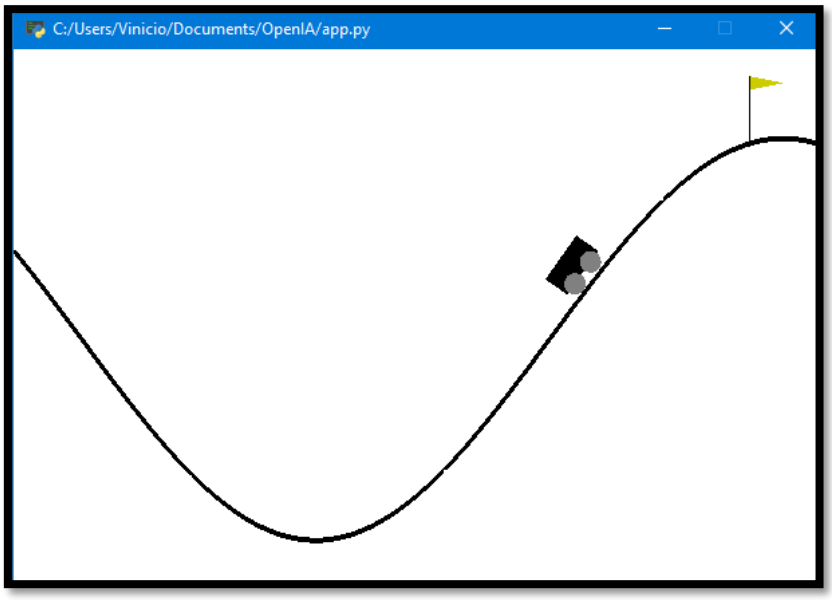
- Usuo con Python
  - Clonanos Universe de openAI desde su repositorio:
    - **git clone** https://github.com/openai/universe.git
~~~~~~

    - **cd universe**
    - **pip install -e .**
- Clonamos librería de GYM
    - **git clone https://github.com/openai/gym.git**
    - **cd gym**
    - **pip install -e .**

[Nota]: En caso de errores leer el README.rts que incluye cada proyecto clonado.

- **Test: Vehículo calcula la velocidad para llegar a la meta**

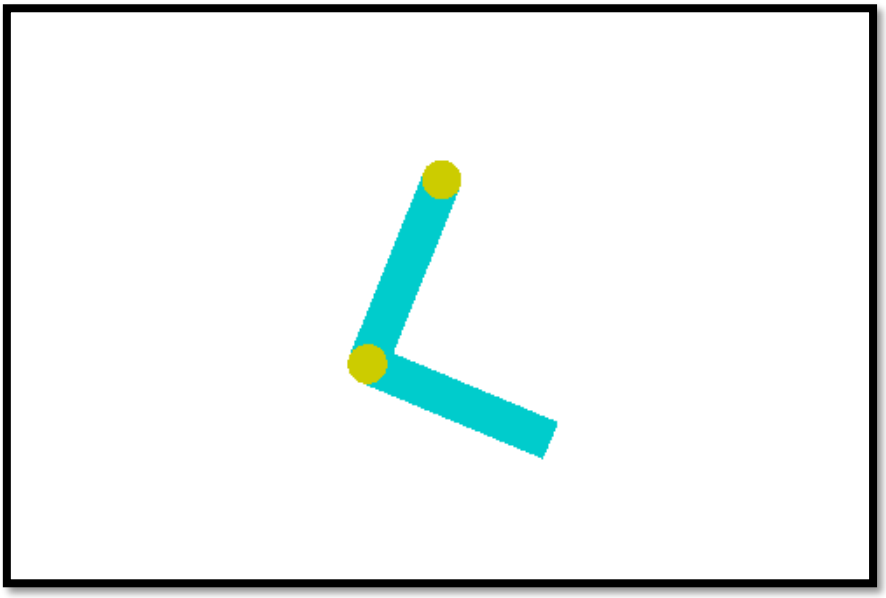
```
import gym
env = gym.make('MountainCarContinuous-v0')
env.reset()
for _ in range(5000):
    env.render()
    env.step(env.action_space.sample()) # take a random action
env.close()
```



Entre más tiempo le damos al sistema, el vehículo llega a tener una mayor aceleración.

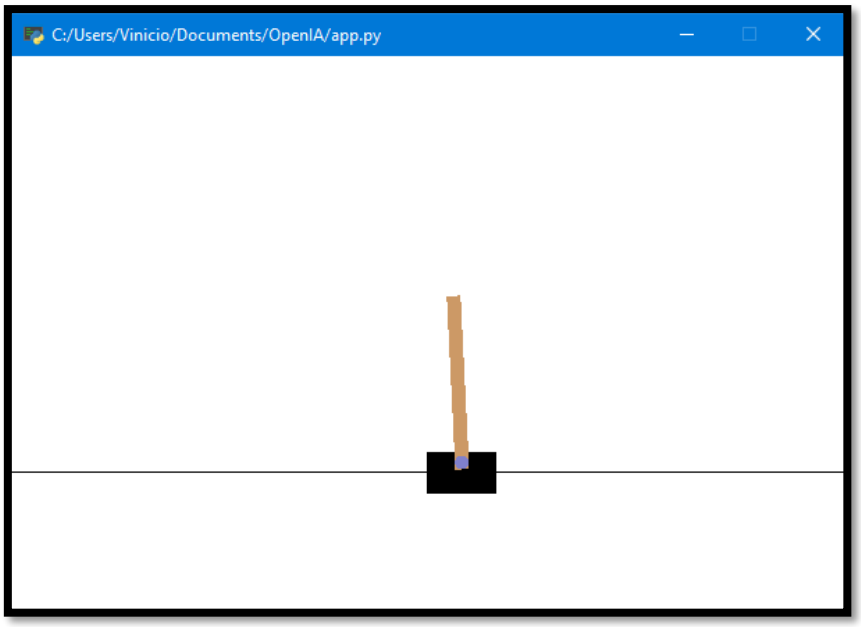
- **Test: Péndulo, Giro completo, en mis pruebas si le doy un rango de 9000 logra dar la vuelva.**

```
import gym
env = gym.make('Acrobot-v1')
env.reset()
for _ in range(9000):
    env.render()
    env.step(env.action_space.sample()) # take a random action
env.close()
env = gym.make('CartPole-v0')
```



- **Test: Seguir la linea, con un tiempo de 100 llegar a recorrer mas las linea**

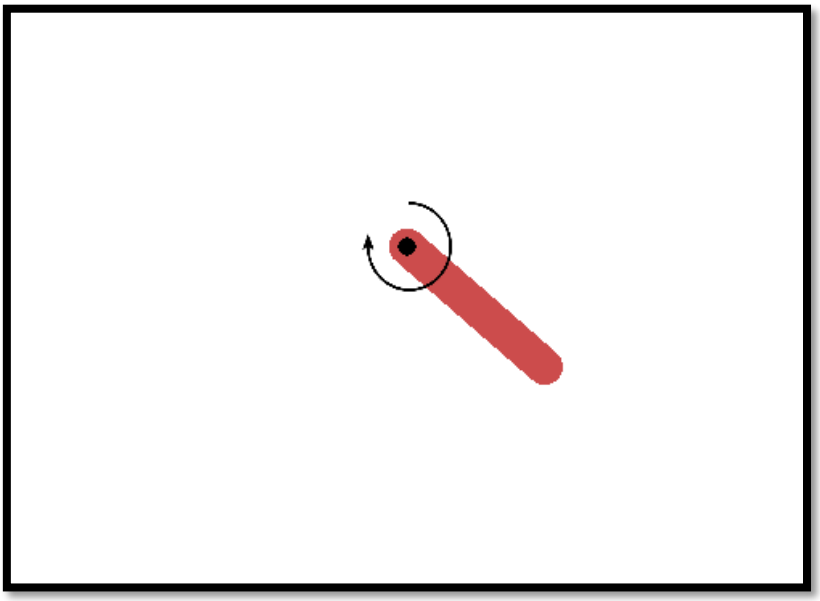
```
import gym
env = gym.make('CartPole-v0')
for i_episode in range(20):
    observation = env.reset()
    for t in range(100):
        env.render()
        print(observation)
        action = env.action_space.sample()
        observation, reward, done, info = env.step(action)
        if done:
            print("Episode finished after {} timesteps".format(t+1))
            break
env.close()
```



- Giro completo, Entrenamiento para que objeto de giro completo

```
import gym
env = gym.make('Pendulum-v0')
env.reset()
for _ in range(1000):
    env.render()
```

```
    env.step(env.action_space.sample()) # take a random action
env.close()
```



OpenAI Universe

- Universe incluye los juegos Atari 2600 del Arcade Learning Environment.
- Universe incluye 1,000 juegos Flash (100 con funciones de recompensa), que distribuimos en la imagen de Docker quay.io/openai/universe.flashgames con el consentimiento de los titulares de derechos.

```
import gym

env = gym.make('flashgames.DuskDrive-v0') # any Universe environment ID here
observation_n = env.reset()

while True:
  # agent which presses the Up arrow 60 times per second
  action_n = [[('KeyEvent', 'ArrowUp', True)] for _ in observation_n]
  observation_n, reward_n, done_n, info = env.step(action_n)
  env.render()
```

- **Conclusiones**

- Los entrenamientos de openIA son muy prácticos para que los desarrolladores puedan practicar y entrenar más sistemas copases de realizar tares simples que los humanos pueden desarrollar.

- Tener cuidado con el uso de los sitemas entrenados, ya que con el uso de GPT-3 se puede llegar a generar noticias falsas, pero siendo 100% creibles.

- **Referencias**

[1] RANA, A. (21 de Sep de 2018). https://towardsdatascience.com/. Obtenido de https://towardsdatascience.com/: https://towardsdatascience.com/reinforcement-learning-with-openai-d445c2c687d2

[2] Erin Pettigrew. (2016). Universe. 2020, de OpenIA Sitio web: https://openai.com/blog/universe/