

Instrucciones para utilizar el entorno *Football*

Football es un entorno para aprendizaje por refuerzo profundo ofrecido por Google Research compatible con OpenAI Gym.



Este entorno simula un partido de fútbol, con las reglas habituales. Los jugadores, humanos o agentes, pueden manejar a los jugadores y realizar diversas acciones dependiendo del estado del juego.

Puede encontrar información detallada sobre *Football* aquí:

<https://ai.googleblog.com/2019/06/introducing-google-research-football.html>

Se recomienda leer el siguiente artículo, que describe las principales características de *Football*.

<https://github.com/google-research/football/blob/master/paper.pdf>

La descripción completa y el código fuente están disponibles en GitHub:

<https://github.com/google-research/football>

Football ofrece varias configuraciones alternativas a la de partido completo, que son útiles para probar agentes en un entorno más reducido. En concreto, se propone la utilización de "3 vs 1 with Keeper", que simula un ataque de 3 jugadores contra el portero con el objetivo de marcar gol.

Instalar simulador

A continuación, se proporcionan instrucciones para instalar *Football*.

Notación:

<comandos en terminal>

0. Pre-requisitos

- Ubuntu 18.04
- Computador con tarjeta gráfica con soporte para versión de OpenGL >= 3.3 y drivers actualizados

Puede ser una máquina virtual con VMWare. Parallels no soporta OpenGL >= 3 y, por lo tanto, no puede utilizarse. Para actualizar OpenGL:

```
sudo add-apt-repository ppa:ubuntu-x-swat/updates  
sudo apt-get dist-upgrade
```

1. Instalar Anaconda

Descargar Anaconda:

```
wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86_64.sh
```

Instalar (con las opciones por defecto):

```
sh Anaconda3-2019.03-Linux-x86_64.sh
```

Reiniciar terminal y lanzar navegador de Anaconda:

```
anaconda-navigator
```

2. Instalar dependencias

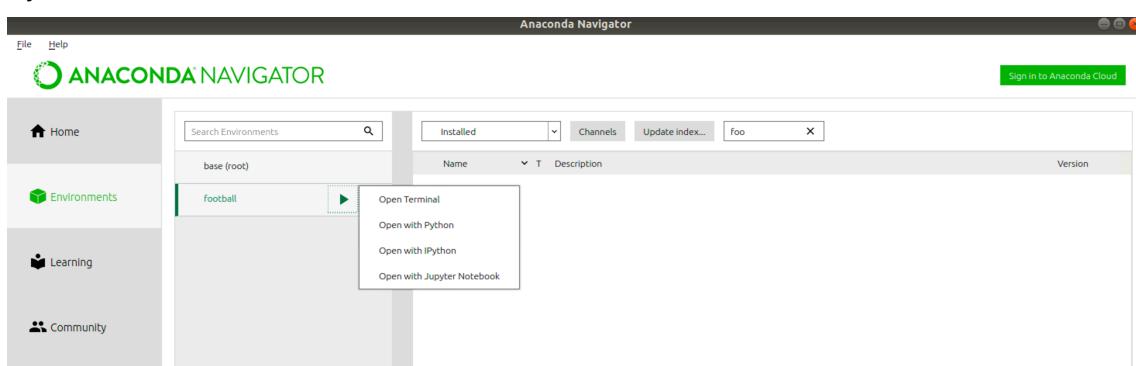
```
sudo apt-get install git cmake build-essential libgl1-mesa-dev libsdl2-dev  
libsdl2-image-dev libsdl2-ttf-dev libsdl2-gfx-dev libboost-all-dev  
libdirectfb-dev libst-dev mesa-utils xvfb x11vnc libsqlite3-dev glee-dev  
libsdl-sge-dev python3-pip libosmesa6-dev libgl1-mesa-glx libglfw3
```

Este paso puede requerir reparar instalaciones incompletas. En ese caso, ejecutar:

```
sudo apt --fix-broken install  
sudo apt-get update
```

3. Crear entorno *virtualenv*

En el navegador de Anaconda, crear un nuevo entorno llamado *football* basado en Python 3.7:



Abrir en nuevo entorno en un terminal con *Environments > football > Open Terminal*. Los siguientes comandos deben ejecutarse en este mismo terminal, con el entorno *football* que acabamos de crear en Anaconda activado.

4. Instalar paquetes Python requeridos

Instalar Tensorflow en el entorno *football*:

```
pip install tensorflow      # sin GPU  
pip install tensorflow-gpu # con GPU
```

Instalar OpenAI baselines:

```
git clone https://github.com/openai/baselines.git  
cd baselines  
pip install -e .
```

5. Instalar *Football*

Salir de la carpeta anterior e instalar el propio entorno:

```
git clone https://github.com/google-research/football.git  
cd football  
pip install .[tf_cpu]           # usar [tf_gpu] si GPU
```

6. Comprobar entorno

Lanzar juego

Desde la carpeta football, ejecutar:

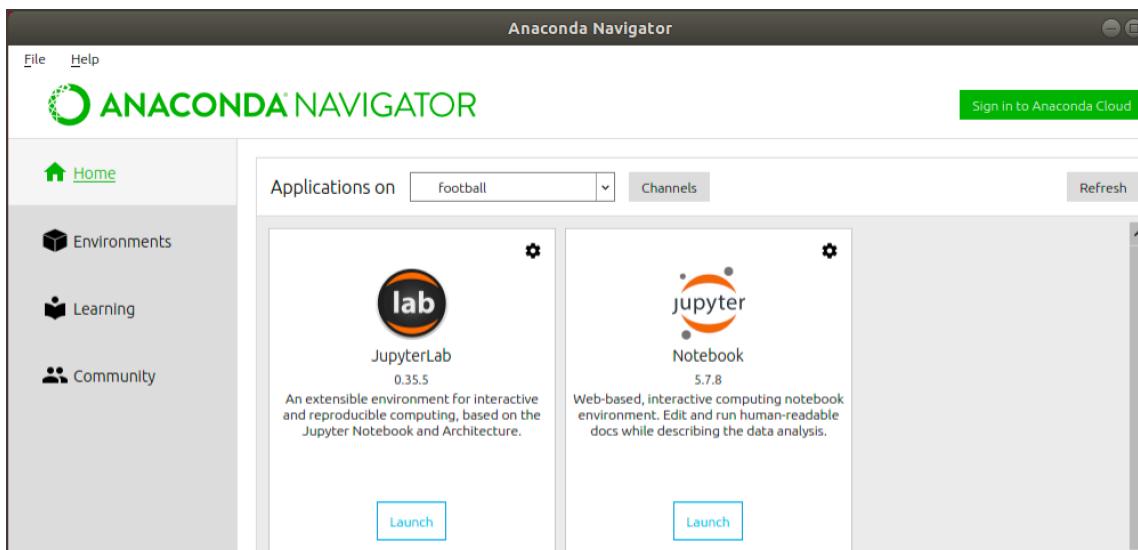
```
python -m gfootball.play_game
```



Jugar con jugador aleatorio

Abrir JupyterLab para el entorno *football* de Anaconda.

Cargar el cuaderno `football-random-player.ipynb` y ejecutar.



Ejecutar ejemplo de entrenamiento con PPO2:

```
python3 -m gfootball.examples.run_ppo2 --level=academy_empty_goal_close
```

7. Implementar nuestro agente

Recordar: es necesario instalar PyTorch y otras dependencias utilizadas en las implementaciones de este curso.

```
pip install torch torchvision numpy matplotlib
```

El entorno incluye varias implementaciones de jugadores:

<https://github.com/google-research/football/tree/master/gfootball/env/players>

En este ejercicio implementaremos un agente capaz de resolver el escenario del entorno *Football*. Llamado `academy_empty_goal_close`. El objetivo es entrenar al agente para que consiga la mayor puntuación considerando *checkpoints*.