### Laboratorio: Trabajo en remoto con VSCode



Grado en Ciencia e Ingeniería de Datos (Universidad de Oviedo)

Pablo González, Pablo Pérez {gonzalezgpablo, pabloperez}@uniovi.es Centro de Inteligencia Artificial, Gijón

#### Introducción

Cuando trabajamos en aprendizaje profundo, es muy normal que las máquinas donde lancemos los trabajos no estén físicamente donde estamos nosotros. Normalmente además se tratará de servidores **Linux** con una o más **tarjetas gráficas instaladas**.

Hay una serie de herramientas que nos facilitan el trabajo en remoto. Por citar algunas:

- USCode (permite a través de plugins el trabajo en remoto)
- Screen (útil para lanzar entrenamientos largos)

### Herramientas para el trabajo en remoto

La herramienta principal utilizada para el trabajo en remoto es **SSH**. Es una herramienta muy potente que nos permitirá acceder a una consola de la máquina así como a redireccionar puertos a través de túneles para el acceso a servicios de la misma.

# ¿Qué es SSH?

### ¿Qué es SSH?

- SSH (Secure Shell) es un protocolo de red que permite a los usuarios acceder y gestionar de forma segura dispositivos y sistemas remotos.
- Proporciona un canal seguro de comunicación sobre redes inseguras, como Internet, utilizando técnicas de cifrado para proteger los datos transmitidos, reemplazando a herramientas como Telnet, que envían información sin cifrar, lo que representa un riesgo para la seguridad.



## ¿Como conectarse?

Necesitamos un **cliente de SSH**. En linux ya lo tenemos instalado por defecto en nuestra consola. En Windows, si no tenemos powershell, podemos utilizar **Putty**.

ssh user@machineip

#### Autenticación

Podemos realizar la autenticación usando una **password** tradicional o usando un mecanismo de **clave pública y privada**.

#### **VSCode**

Más adelante veremos que VSCode tiene su propio cliente de SSH integrado.

## Comandos útiles [htop]

0.00 B/s 28439 pgonzalez F1Help F2Setup F3SearchF4F3

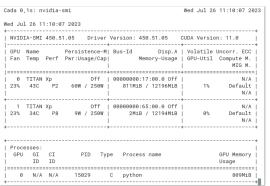
El primer comando útil en un servidor para conocer su estado general es **htop**. Con él podemos ver el estado de los procesos, manejarlos, la memoria, entre otros muchos datos.

```
10.9G/62.6Gl
                                                            Tasks: 132, 573 thr; 19 running
Swp[||||||||||||||
                                             7.496/31.061
                                                            Load average: 18.00 18.00 17.18
                                                            Uptime: 309 days(!), 23:23:24
 no perm 14153 quevedo
                                                                 5:13.25 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14171 guevedo
                                                                 4:45.22 python 1B HacerExperimentosEvaluacion.pv .//Folds
 no perm 14298 quevedo
                                         570M 54104 R 100.
                                                                 1:46.76 pvthon 1B_HacerExperimentosEvaluacion.pv .//Folds
                                                                 2:03.37 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14271 quevedo
                                                                 0:57.30 python 1B HacerExperimentosEvaluacion.pv .//Folds
 no perm 14350 guevedo
                                         408M 53640 R 100.
 no perm 14378 quevedo
                                         352M 54232 R 100.
                                                                 0:42.58 python 1B HacerExperimentosEvaluacion.pv .//Folds
                                                                 5:14.62 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14144 quevedo
 no perm 14135 quevedo
                                                                 5:24.02 python 1B_HacerExperimentosEvaluacion.py .//Folds
                                0 1073M
                                         755M 54080 R 99.8
 no perm 14307 guevedo
                                                                1:46.37 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14289 quevedo
                                         576M 54336 R 99.8
                                                                1:54.42 python 1B HacerExperimentosEvaluacion.pv .//Folds
 no perm 14280 quevedo
                                   932M
                                                                 1:54.87 python 1B_HacerExperimentosEvaluacion.py .//Folds
                                                                 0:43.10 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14369 quevedo
 no perm 13977 guevedo
                                                           1.2 9:54.77 python 1B HacerExperimentosEvaluacion.pv .//Folds
                                                                 0:33.34 python 1B_HacerExperimentosEvaluacion.pv .//Folds
 no perm 14387 quevedo
                                                           0.5 0:27.46 python 1B_HacerExperimentosEvaluacion.py .//Folds
 no perm 14396 quevedo
                                                           1.1 11:28.11 python 1B HacerExperimentosEvaluacion.pv .//Folds
 no perm 13955 guevedo
                                                            1.2 5:03.09 python 1B HacerExperimentosEvaluacion.pv .//Folds
 no perm 14162 quevedo
0.00 B/s 14368 pgonzalez
                                              3196 R
                                                            0.0 0:00.68 htop
```

179M 25020 S 1.3 0.3 22:04.89 /home/pgonzalez/.dropbox-dist/dropbox-lnx.x86\_64-

## Comandos útiles [nvidia-smi]

Otro comando muy útil es **nvidia-smi**. Con él podemos ver el estado de la tarjeta gráfica. La memoria ocupada, su utilización, temperatura, etc.



#### Comando watch

El comando **watch** se usa en combinación con nvidia-smi. Permite ver la salida del comando cada cierto tiempo: watch —n 0,1 nvidia—smi.

### Usando VSCode

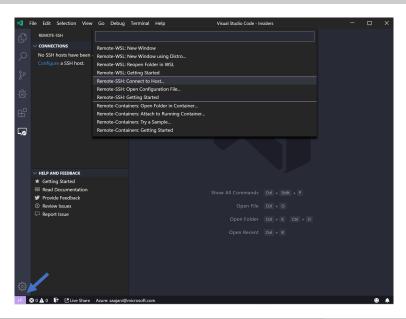
Usando la extensión **Remote - SSH (ms-vscode-remote.remote-ssh)** podemos abrir una conexión remota por SSH sobre cualquier máquina a la que tengamos acceso. Esto nos permite:

- Abrir y almacenar archivos en el almacenamiento del servidor.
- Ejecutar y depurar programas usando los recursos del servidor.
- Crear redirecciones de puertos automáticamente para servicios lanzados en el servidor.
- Utilizar terminales remotas sin necesidad de ningún programa extra.

#### VSCode-server

También existe la opción de correr de manera íntegra VSCode en el servidor y acceder a él a través de un navegador web. Para ello tendremos que instalar https://code.visualstudio.com/docs/remote/vscode-server.

#### Usando VSCode



# Lanzando procesos [nohup y screen]

Generalmente los entrenamientos en el aprendizaje profundo son procesos que están pueden llegar a ejecutarse un largo tiempo. Es importante conocer como lanzarlos para que Linux no los termine cuando cerremos la terminal o el equipo desde el que estamos trabajando. Las dos mejores opciones para ello son:

- **nohup**. nohup comando [opciones] &. Se utiliza para ejecutar un programa o comando en segundo plano y evitar que se detenga cuando se cierra la terminal o sesión de inicio de la que se lanzó.
- screen. Se utiliza para crear y administrar sesiones de terminal virtuales. Estas sesiones son como ventanas separadas en las que puedes ejecutar comandos o programas de forma independiente y luego volver a conectarte a ellas en cualquier momento, incluso después de cerrar la sesión original. Una guía rápida de screen se puede encontrar aquí.