Tutorial

Entorno docker milvus

Propósito

El propósito de este tutorial es enseñar como usar el entorno docker para trabajar con Milvus, así como configurar el docker para tener acceso a la GPU del equipo.

Entorno de trabajo

El entorno se puede descargar desde el ***Moodle*** de la asignatura y viene comprimido en un fichero .zip. Una vez descomprimido, se creará la siguiente estructura de carpetas.

- **dockerimg:** Contiene la imagen del entorno jupyter
    - Dockerfile: Contiene las instrucciones para instalar el entorno jupyter.
- **jupyter_volume:** Aquí se almacenan los ficheros que podemos crear y edirar desde el jupyter lab.
- **milvus_volumes:** Aqui almacena milvus la información de nuestra base de datos vectorial
- ***docker-compose.yml:*** fichero con la configuración de los contenedores.
- ***.env:*** fichero con valores de configuración.
    - USER=<nombre de tu usuario, en linux se puede ver con el comando “whoami”>
    - UID=<id de tu usuario, en linux se puede ver con el comando “id -u”>
    - GID= =<id del grupo tu usuario, en linux se puede ver con el comando “id -g”>
    - JUPYTER_PORT=<puerto en donde se abrirá jupyter lab>

Arrancar el entorno

1. Abrir un terminal en la carpeta donde habéis descomprimido el entorno. En windows se puede hacer escribiendo powershell en el recuadro de la ruta del navegador de ficheros.
2. Ejecutar “`docker-compose –d`".
3. Comprobar que están arrancados los servicios de “*milvus-standalone”*, *“minio”*, *“etcd”* y *“notebook”* ejecutando el comando "`docker-compose ps`”.
4. Podrás acceder a jupyer desde el navegador introduciendo la URL *“localhost:{JUPYTER_PORT}”* por defecto es el puerto 8899.
    a. Si jupyter te pide un token, puedes obtenerlo ejecutando el siguiente comando en un terminal abrierto en la carpeta del entorno “`docker-compose logs notebook | grep token`”
5. Podremos conectarnos a Milvus desde python en el host *“milvus-standalone”* y el puerto por defecto *“19530”*.

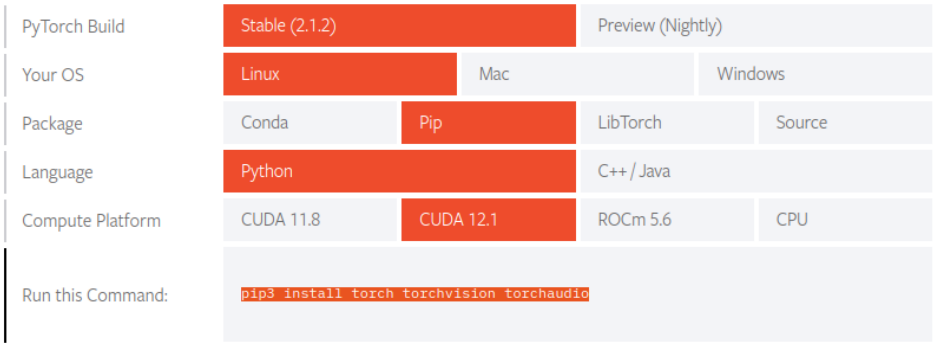
Uso de GPUs nvidia y CUDA

No es necesario para la práctica, pero si se cuenta con una GPU de Nvidia el contendor docker puede acceder a ella para acelerar el cálculo de los embeddings. Para ello es necesario usar la imagen de nvidia que se corresponda con los drivers de nuestra GPU. Ejecuta en un terminal el comando *`nvidia-smi`* y apunta el valor de "*CUDA Version*".

1. Edita el fichero *Dockerfile* para introducir tu versión de CUDA en la cabecera.

   ia/cuda:**12.2.2**-devel-ubuntu22.04

2. Dependiendo de tu versión de CUDA es posible que se necesite un comando especial para instalar pytorch. Para saber si es necesario, visita la siguiente página web https://pytorch.org/get-started/locally/ y marca tu versión de CUDA, sistema LINUX y PACKAGE linux.



   a. Si el comando que aparece es *"`pip3 install torch torchvision torchaudio`"*, entonces no es necesario cambiar nada.
   b. En otro caso es necesario cambiar el comando para instalar pytorch del Dockerfile por el que te indiquen en la web

   ...
   USER root
   RUN pip3 install torch \\ comando que debo cambiar
   ...

3. Una vez realizados los cambios, hay que construir de nuevo el contenedor ejecutando el comando "`docker-compose build`".