DL in Docker on Windows

LDL / By magnus

(This blog post represents my personal opinion, and should not be interpreted as any official statements from my employer NVIDIA.)

Microsoft's Windows Subsystem for Linux 2 (WSL2) enables NVIDIA's AI stack to run on Windows. In this blog post I will describe how to use WSL2 to run the LDL code examples in a Docker container.

First install WSL2 and Docker by following the following instructions: Getting Started with NVIDIA's AI Platform on GeForce RTX PCs

Assuming that everything went according to plan, you should now be able to run a Docker image obtained from NVIDIA GPU Cloud (ngc). Try with the following command line:

```
sudo docker run --gpus all -it --rm
nvcr.io/nvidia/tensorflow:22.10-tf2-py3
```

Note that the 22.10 in the command line above (and in the command lines further down) refer to the version (2022 October), so you might want to update to something newer depending on when you read this. Once you have confirmed that the command line above works, exit the Docker container:

```
exit
```

In addition to the DL framework itself, you also need the code examples that you want to run. They can be downloaded as a zip file from here: https://github.com/NVDLI/LDL/archive/refs/heads/main.zip

Go to a suitable location, and download and unzip it with the following commands:

```
wget https://github.com/NVDLI/LDL/archive/refs/heads/main.zip
unzip LDL-main.zip
```

The command lines below assume that it is placed in the following location:

```
/home/USERNAME/LDL-main
```

We can now start Docker with the following command line:

```
sudo docker run --gpus all --shm-size=1g --ulimit memlock=-1 -
-ulimit stack=67108864 --rm -it -p 8888:8888 -v
/home/USERNAME/LDL-main:/home/LDL
nvcr.io/nvidia/tensorflow:22.10-tf2-py3 /bin/bash
```

In this example we use TensorFlow. See further down for PyTorch.

Some additional information in case you are interested in the details:

- The options "–shm-size=1g –ulimit memlock=-1 –ulimit stack=67108864" are recommended to use with these images. If you don't include these options on the command line, then you will get a warning message.
- The option "-p 8888:8888" has to do with port forwarding to be able to run Jupyter notebooks. You can omit this option if you just want to run the normal Python files.
- The option "-v /home/USERNAME/LDL-main:/home/LDL" mounts the directory /home/USERNAME/LDL-main in the location /home/LDL inside the Docker container.

The Docker images do not have matplotlib or idx2numpy pre-installed so if you want to run code examples that rely on these modules, then do the following:

```
pip install matplotlib
pip install idx2numpy
```

Let's now try a code example:

```
cd /home/LDL/tf_framework
python c6e1_boston.py
```

If you instead want to run it as a Jupyter notebook, then do the following:

```
cd /home/LDL
jupyter notebook --ip 0.0.0.0 --port 8888 --allow-root
```

You should see something along the following lines being printed in your shell:

```
To access the notebook, open this file in a browser:
file:///root/.local/share/jupyter/runtime/nbserver-362-
open.html

Or copy and paste this URL:
http://hostname:8888/?
token=80d3a14a3660b254c26e3f983c8cf26e3de0714de3b9830e
```

You can now copy the URL above and paste it into a browser, but replace "hostname" with "localhost". This should enable you to run the notebooks in your browser.
If you instead want to run PyTorch, you would use the following Docker command line:

```
sudo docker run --shm-size=1g --ulimit memlock=-1 --ulimit
stack=67108864 --rm -it -p 8888:8888 -v
/home/magnus/projects/LDL-main:/home/LDL
nvcr.io/nvidia/pytorch:22.10-py3 /bin/bash
```

However, some of the code examples use some utilities from TensorFlow so you will also need to install TensorFlow in your container:

```
pip install tensorflow
```

I have also run into some problems with Jupyter notebooks if it is not upgraded to a newer version:

```
pip install --upgrade notebook
```

And as before, we need matplotlib and idx2numpy:

```
pip install matplotlib
pip install idx2numpy
```

Try a code example:

```
cd /home/LDL/pt_framework
python c6e1_boston.py
```

Or if you want to run it as a Jupyter notebook, then do the following as described above:

```
cd /home/LDL
jupyter notebook --ip 0.0.0.0 --port 8888 --allow-root
```

Extending a Docker Image

In the examples above, we had to install some missing packages to the Docker image. Instead of doing that each time, you can create your own new Docker image based on a base image. This is done by using a Dockerfile. The LDL repository contains two different files:

- Dockerfile_tf – if you want to run the TensorFlow examples
- Dockerfile_pt – if you want to run the PyTorch examples

You can create your image using one of the following two commands, depending on if you want to run TensorFlow or PyTorch:

```
sudo docker build -t ldl_tf:v1 -f Dockerfile_tf .
sudo docker build -t ldl_pt:v1 -f Dockerfile_pt .
```

This will create a new image named ldl_tf:v1 (TensorFlow) or ldl_pt:v1 (PyTorch). You can now start Docker with one of the following two commands, and you will no longer need to install additional packages before running the programming examples:

```
sudo docker run --gpus all --shm-size=1g --ulimit memlock=-1 -
-ulimit stack=67108864 --rm -it -p 8888:8888 -v
/home/USERNAME/LDL-main:/home/LDL ldl_tf:v1 /bin/bash

sudo docker run --gpus all --shm-size=1g --ulimit memlock=-1 -
-ulimit stack=67108864 --rm -it -p 8888:8888 -v
/home/USERNAME/LDL-main:/home/LDL ldl_pt:v1 /bin/bash
```

Hopefully this is helpful. If you find anything that looks wrong, then please submit a report on the errata page: https://ldlbook.com/errata/

← Previous Post Next Post →

Home Downloads Errata Blog