Python and GPU setup

Compiled by: Prof Eduan Kotzé

August 27, 2024

Contents

1 Introduction

This booklet will be used as background knowledge to setup Python with a Graphical Processing Unit (GPU). It was compiled using TeXstudio and LaTeX.

2 NVIDIA Installation

Start with downloading the latest drivers from NVIDIA for your GPU.
Visit: `https://www.nvidia.com/download/index.aspx`
And select the appropriate driver for your NVIDIA product.
I logged in with my Google Account to downloaded the driver version 552.12.

3 Python Installation

- Visit Python Website
  `https://www.python.org/downloads/`

For my machine, I installed **Python 3.10.11 (64-bit)**. According to their documentation, 3.10.11 was the last full bugfix release of Python 3.10 with binary installers.

Download from: `https://www.python.org/downloads/release/python-31011/`

4 Tensorflow-GPU

Tensorflow (Abadi et al., 2015) GPU support on **native-Windows is only available for 2.10 or earlier versions**. Starting in TF 2.11, CUDA build is not supported for Windows. For using TensorFlow GPU on Windows, you will need to build/install TensorFlow in WSL2 or use tensorflow-cpu with TensorFlow-DirectML-Plugin.

See: `https://www.tensorflow.org/install/source_windows#gpu`

tensorflow-gpu-2.10.1 requires Python version 3.7-3.10, cuDNN 8.1 and CUDA 11.2

NOTE: Since I no longer use Tensorflow (due to the WLS2 issue), I did not install it on my computer.

5 PyTorch

Currently, PyTorch (Paszke et al., 2017) on Windows only supports Python 3.8-3.11; Python 2.x is not supported. See: `https://pytorch.org/get-started/locally/#windows-python` for more details on which Python versions are supported by PyTorch. This also influenced my decision to use Python 3.10.11.

5.1 Prerequisites

Make sure your your machine has a CUDA-enabled GPU.
Visit: `https://developer.nvidia.com/cuda-gpus`.

5.2 Install CUDA Toolkit

I also installed **CUDA Toolkit 12.1** on my machine.
Visit: `https://developer.nvidia.com/cuda-downloads/`

The reason why I installed CUDA Toolkit 12.1 is related to which CUDA version is supported with PyTorch. Consult the following website: `https://pytorch.org/get-started/locally/`

At the time of writing this manual only **CUDA 11.8** and **CUDA 12.1** was supported on Windows 11.

I downloaded from: `https://developer.nvidia.com/cuda-12-1-0-download-archive?target_os=Windows`. The download file was 3.1GB.

5.3 Checking version

Once you installed the CUDA Toolkit, its easy to verify which version and where it was installed with the following command in Jupyter Notebook. Just search for '**where.exe**' on the internet and download it into your working Notebook directory.

```
!where cud*
```

This should produce similar to the following output:

```
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.1\bin\cudafe++.exe
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.1\bin\cudart64_12.dll
```

Follow this up with the following command in Jupyter Notebook:

```
!nvcc --version
```

This should produce similar to the following output:

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2023 NVIDIA Corporation
Built on Wed_Feb__8_05:53:42_Coordinated_Universal_Time_2023
Cuda compilation tools, release 12.1, V12.1.66
Build cuda_12.1.r12.1/compiler.32415258_0
```

5.4 Checking GPU

Next step, see the status of your GPU using the following command in Jupyter Notebook.

```
!nvidia-smi
```

This should produce similar to the following output:

```
Fri Apr 19 13:40:30 2024
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 552.12                 Driver Version: 552.12         CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                     TCC/WDDM  | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA GeForce RTX 4090      WDDM  |   00000000:01:00.0  On |                  Off |
| 39%   74C    P2            367W /  450W |   23199MiB /  24564MiB |     88%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
```

5.5 Install PyTorch with Cuda

To install PyTorch via pip, and do have a CUDA-capable system, in the above selector, choose OS: Windows, Package: Pip and the CUDA version suited to your machine. Often, the latest CUDA version is better. See: `https://pytorch.org/get-started/locally/#with-cuda-1`.

Install with the following command:

```
pip install torch==2.2.2 torchvision --index-url https://download.pytorch.org/whl/cu121
```

5.6 Testing PyTorch environment

Now we are ready to test PyTorch with GPU (CUDA) support. The following are code snippets from Jupyter Notebook.

```
import torch

print(torch.__version__)
2.2.2+cu121
# --> notice the+cu121 indicating its CUDA/GPU supported.

print(torch.cuda.is_available())
True

print(torch.cuda.device_count())
1

print(torch.cuda.get_device_name(0))
'NVIDIA GeForce RTX 4090'
```

5.6.1 Quick PyTorch Test

```
import torch
x = torch.rand(5, 3)
print(x)

tensor([[0.5587, 0.5857, 0.8586],
[0.8116, 0.0824, 0.2300],
[0.8865, 0.9970, 0.3152],
[0.7559, 0.1862, 0.1720],
[0.1194, 0.1675, 0.5612]])
```

6 Huggingface

Once Python, CUDA, and PyTorch is installed, installing Huggingface (Wolf et al., 2020) is straightforward. Install with the following command:

```
pip install datasets transformers tokenizers accelerate
```

7 TRL - Transformer Reinforcement Learning

TRL (von Werra et al., 2020) is a full stack library where we provide a set of tools to train transformer language models with Reinforcement Learning, from the Supervised Fine-tuning step (SFT), Reward Modeling step (RM) to the Proximal Policy Optimization (PPO) step. The library is integrated with transformers. Install with the following command:

```
pip install trl
```

See also: `https://github.com/huggingface/trl`

8 Summary of selected Python libraries installed

List of current libraries (and versions) on my machine (2024/07/31):

- Python: 3.10.11
- torch: 2.2.2+cu121
- transformers (huggingface): 4.42.4
- tokenizers (huggingface): 0.19.1
- datasets (huggingface): 2.19.0
- trl (huggingface): 0.9.6
- scikit-learn: 1.4.2.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... Zheng, X. (2015). *[tensorflow] TensorFlow: A system for large-scale machine learning.* Retrieved from `tensorflow.org`

Paszke, A., Chanan, G., Lin, Z., Gross, S., Yang, E., Antiga, L., & Devito, Z. (2017). [pytorch] Automatic differentiation in PyTorch. In *31st conference on neural information processing systems.*

von Werra, L., Belkada, Y., Tunstall, L., Beeching, E., Thrush, T., Lambert, N., & Huang, S. (2020). *[trl] TRL: Transformer Reinforcement Learning.* \url{https://github.com/huggingface/trl}. GitHub.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. (2020). [huggingface] Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations* (pp. 38–45). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from `https://www.aclweb.org/anthology/2020.emnlp-demos.6` doi: 10.18653/v1/2020.emnlp-demos.6