Python, CUDA and GPU setup Version: 2025.08.12

Compiled by: Prof. Eduan Kotzé

August 13, 2025

Contents

1 Introduction

This booklet serves as a guide to setup Python with a Graphical Processing Unit (GPU).

2 NVIDIA Drivers 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 prefer using the NVIDIA app for **automatic** driver updates. You can download the app from: `https://www.nvidia.com/en-us/geforce/drivers/`

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. Although not the latest version of Python, it is *my* preferred version for compatibility.
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

At the time of writing this manual (August 12, 2025), PyTorch (Paszke et al., 2017) on Windows only supports Python 3.9-3.12; 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.

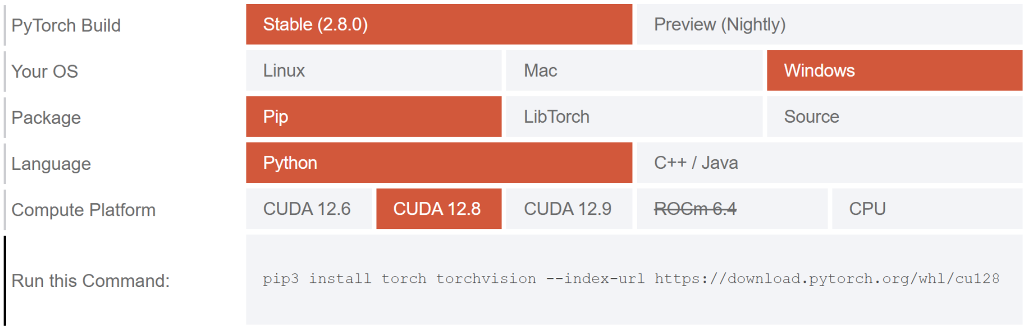
5.1 Prerequisites

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

5.2 Install CUDA Toolkit

You also need to install the CUDA toolkit. I installed **CUDA Toolkit 13** on my machine. I downloaded from:
https://developer.nvidia.com/cuda-downloads?target_os=Windows&target_arch=x86_64&target_version=11&target_type=exe_local.
The download file was 2.3GB.

As of the date of writing this manual (August 12, 2025), **CUDA 12.6**, **CUDA 12.8** and **CUDA 12.9** were supported on Windows 11 for the latest version of PyTorch 2.8.0. Consult the following website for latest versions: https://pytorch.org/get-started/locally/



PyTorch Build	[✓ Stable (2.8.0)]	Preview (Nightly)			
Your OS	Linux	Mac	[✓ Windows]		
Package	[✓ Pip]	LibTorch	Source		
Language	[✓ Python]	C++ / Java			
Compute Platform	CUDA 12.6	[✓ CUDA 12.8]	CUDA 12.9	~~ROCm 6.4~~	CPU

Run this Command:

```
pip3 install torch torchvision --index-url https://download.pytorch.org/whl/cu128
```

Figure 1: PyTorch + CUDA support.

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.

```
!where cud*
```

This should produce similar to the following output:

```
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v13.0\bin\x64\cudart64_13.dll
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v13.0\bin\cudafe++.exe
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.9\bin\cudafe++.exe
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.9\bin\cudart64_12.dll
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common\cudart32_65.dll
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common\cudart64_65.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) 2005-2025 NVIDIA Corporation
Built on Wed_Jul__6_20:06:48_Pacific_Daylight_Time_2025
Cuda compilation tools, release 13.0, V13.0.48
Build cuda_13.0.r13.0/compiler.3620728_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:

```
Tue Aug 12 17:32:26 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 576.02                 Driver Version: 576.02         CUDA Version: 12.9     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                  Driver-Model | 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 3060      WDDM  |   00000000:01:00.0  On |                  N/A |
|  0%   40C    P8             18W /  170W |     347MiB /  12288MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
```

5.5 Install PyTorch with Cuda

To install PyTorch via pip go to the following url: https://pytorch.org/get-started/locally/ and select the same options as demonstrated in Figure 1. Install with the command:

```
pip3 install torch torchvision --index-url https://download.pytorch.org/whl/cu128
```

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 platform

print(platform.python_version())
3.10.11

import torch

print(torch.__version__)
2.8.0+cu128
# --> notice the+cu128 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 Working on the lab PCs

To open a Jupyter notebook on the lab PCs:

1. Open the Command Prompt.
2. Type `jupyter lab`.
3. Press `Enter`.

If you encounter an error, try to run Command Prompt as administrator.

9 Summary of selected Python libraries installed

List of current libraries (and versions) on the lab machines (2025/08/12):

- Python: 3.10.11
- torch: 2.8.0+cu128
- transformers (huggingface): 4.55.0
- tokenizers (huggingface): 0.21.4
- datasets (huggingface): 4.0.0
- trl (huggingface): 0.21.0
- scikit-learn: 1.7.1

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.* Long Beach, California, USA. Retrieved from https://openreview.net/forum?id=BJJsrmfCZ

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