Tensorflow Tutorial

Lesson 0

- Setting up DIE

- Installing CUDA

- Installing cuDNN

- Installing Tensorflow

→ Biased towards Windows, but short notices for Linux (Ubuntu) will be made

Setting up Python-IDE

- Python builds the baseline, for an easy install use Spyder which is integrated in Anaconda:

  - https://www.anaconda.com/download/
  - Choose according to your OS and Python > 3, 64 bit.
  - Good read to distinguish between Python 2 and Python 3: https://wiki.python.org/moin/Python2orPython3
  - Python Cheat Sheet: https://github.com/ehmatthes/pcc/releases/download/v1.0.0/beginners_python_cheat_sheet_pcc_all.pdf

Version Control

- This slide will not give details but recommends either github or gitlab as an freely available version control

  - https://about.gitlab.com/
  - https://github.com/about

CUDA

- CUDA is an SDK for parallelizing code for the GPU, you can only use it with an NVIDIA®-GPU card

    - Check the compute capability of your GPU here:
        - https://developer.nvidia.com/cuda-gpus
    - You can check your GPU with the help of Chrome:
        - Type: chrome:gpu in the browser window
    - You can use a benchmakrk to get all needed details:
        - https://www.techpowerup.com/gpuz/

    - **→ IF YOU DO NOT HAVE A CUDA ENABLED GPU YOU CAN SKIP THE NEXT SLIDES TILL INSTALLING TENSORFLOW !**

CUDA & Drivers

- To install CUDA Drivers:
  - On Windows:
    - I recommend to download „Geforce Experience" which will install needed drivers
    - https://www.nvidia.com/en-us/geforce/geforce-experience/
  - On Ubuntu:
    - Either do it with the help of the „Additional drivers" feature. Note that if you install the CUDA drivers like this, you must not install them again when it comes to downloading and installing CUDA!

CUDA

- To install CUDA, download the packages (Base-Installer) according to your OS here:
  - https://developer.nvidia.com/cuda-downloads

cuDNN

- To download and install you need to create a developer account at NVIDIA
  - https://developer.nvidia.com/rdp/form/cudnn-download-survey

- Download according to your installed CUDA – Version and OS
  - You can check your CUDA version by typing nvcc --version in the command line (May not work under Windows, if nvcc is not in your environment path)

cuDNN

- Unpack the files downloaded

- Copy the following files into the CUDA Toolkit directory.
  - Copy \cuda\bin\cudnn64_7.dll to
    C:\Program Files \NVIDIA GPU Computing Toolkit\CUDA\vX.X\bin
  - Copy \cuda\ include\cudnn.h to
    C:\Program Files \NVIDIA GPU Computing Toolkit\CUDA\vX.X\include
  - Copy \cuda\lib\x64\cudnn.lib to
    C:\Program Files \NVIDIA GPU Computing Toolkit\CUDA\vX.X\lib\x64

- Where X.X is a placeholder for your CUDA-Version

Installing Tensorflow

- With CUDA-Enabled GPU:
  - Start a terminal, and type:
    - pip install tensorflow-gpu
- Without CUDA-Enabled GPU:
  - Start a terminal, and type:
    - pip install tensorflow
- →If pip is not found, direct to C:\[pythondir]\scripts and open the terminal there, or add it to your path.
- →You also can use the anaconda navigator to install the needed packages.
- For a direct install use:
  - https://www.tensorflow.org/install/

Check your Installation

- CUDA:
  - Open the command window and type
    - nvcc --version
    - It should show the version of the CUDA compiler
- cuDNN
  - Linux:
    - cat /usr/local/cuda/include/cudnn.h | grep CUDNN_MAJOR -A 2
  - Windows:
    - …

Check Tensorflow

- For a simple test application, open Spyder and use the following lines of code:

```python
import tensorflow as tf
with tf.device('/gpu:0'):
    a = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[2, 3], name='a')
    b = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[3, 2], name='b')
    c = tf.matmul(a, b)
    with tf.Session() as sess: print (sess.run(c))
```

GPU

```python
import tensorflow as tf
with tf.device('/cpu:0'):
    a = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[2, 3], name='a')
    b = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[3, 2], name='b')
    c = tf.matmul(a, b)
    with tf.Session() as sess: print (sess.run(c))
```

CPU