Efficient and Distributed training with TensorFlow on Piz Daint

Running TensorFlow on Piz Daint

Guilherme Peretti-Pezzi and Rafael Sarmiento
ETHZürich / CSCS

Lugano, 14th-15th March 2019

- Obtaining TF
  - Generic binaries
  - Containers
  - Build from sources
  - Performance
- Getting started at Piz Daint
  - SSH
  - JupyterLab

TensorFlow binaries

CPU and GPU packages

- CPU-only
  - `pip install tensorflow`
  - `conda install tensorflow`
- GPU-enabled
  - `pip install tensorflow-gpu`
  - `conda install tensorflow-gpu`

Disclaimer

- MacOS and Ubuntu are officially supported
- Horovod must be installed appart

Containers

Official images from DockerHub

- Using Docker (local workstation)
  - `docker pull tensorflow/tensorflow`
- Using Shifter-ng
  - `module load shifter-ng`
  - `shifter pull tensorflow/tensorflow`

Containers

Image from NVIDIA GPU Cloud (NGC)

- Using Singularity(3+)
  - `module load singularity`
  - `singularity pull docker://nvcr.io/nvidia/tensorflow:19.02-py3`
  - https://ngc.nvidia.com/catalog/containers/nvidia:tensorflow

Build from sources

Main requirements

- Python >= 3.4
- Bazel - Google’s build and test tool (https://bazel.build/)
- protobuf - Language-neutral, platform-neutral extensible mechanism for serializing structured data
- cuDNN - NVIDIA CUDA Deep Neural Network library
- SWIG - Simplified Wrapper and Interface Generator

Manual build from source

```
#  Configure and build steps
> ./configure
> bazel build ——config=cuda ——copt=—mavx ——copt=—mavx2 ——copt=—mfma\\
             ——copt=—msse4.2 ——copt=—msse4.1\\
             —c opt //tensorflow/tools/pip_package:build_pip_package

# Installing the generated python wheel
> pip install tensorflow—1.12—<...>—linux_x86_64.whl
```

Automated build with EasyBuild

```
# Configure and build steps
# Prepare build environment
> module load daint—gpu EasyBuild —custom
# Search for available recipes (CrayGNU = Piz Daint)
> eb ——search Tensorflow
# Build
> eb TensorFlow—1.12.0—CrayGNU—18.08—cuda—9.1—python3.eb —r
# load module
> module load TensorFlow/1.12.0—CrayGNU—18.08—cuda—9.1—python3
```

[1] CSCS EasyBuild recipes can also be found on Github
https://github.com/eth-cscs/production

Performance of binary packages

Remark

- Always make sure that your build matches your architecture
- Generic builds usually focus on portability and not always performance
- `pip` TensorFlow packages
  - ‘Starting with TensorFlow 1.6, binaries use AVX instructions which may not run on older CPUs’

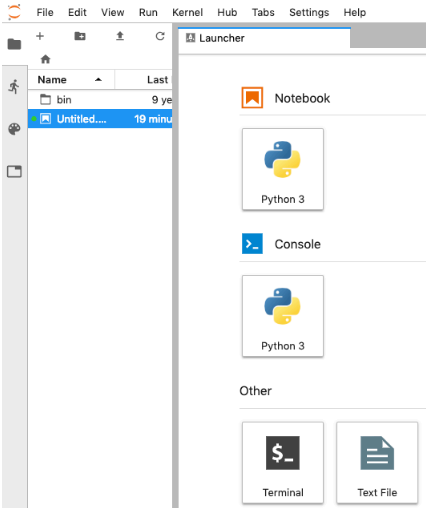
Getting started: Accessing Piz Daint

Hands On on Piz Daint

```
# login
> ssh ela.cscs.ch -l studXX
> ssh daint
# Load GPU software stack & TF module
> module load daint-gpu TensorFlow/1.12.0-CrayGNU-18.08-cuda-9.1-python3

# MNIST example on a single node
> cd $SCRATCH
> wget https://raw.githubusercontent.com/tensorflow/models/master/tutorials/
       image/mnist/convolutional.py
> cp -r /apps/daint/UES/mnist/data/ .
> salloc -N 1 -C gpu --res tensor11
> srun python3 convolutional.py
```

Interactive Supercomputing: JupyterLab interface at CSCS



- Jupyter Notebooks
  - Open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text
- JupyterLab
  - New interface for JupyterHub (Windows & Tabs)
- JupyterHub
  - Multi-user server for Jupyter Notebooks

Steps for running notebooks interactively on Piz Daint

Starting your notebook server

1- Log in using the web interface
  - https://jupyter.cscs.ch

2- Spawn your notebook server
  - This step will grant you exclusive access to a (single or multi-node) job

3- Create/run your notebooks interactively

Steps for running notebooks interactively on Piz Daint

Stopping your server

- **Remember to shutdown your notebook server when you are done!**
- `File` → `Quit`

Importing TensorFlow on JupyterLab

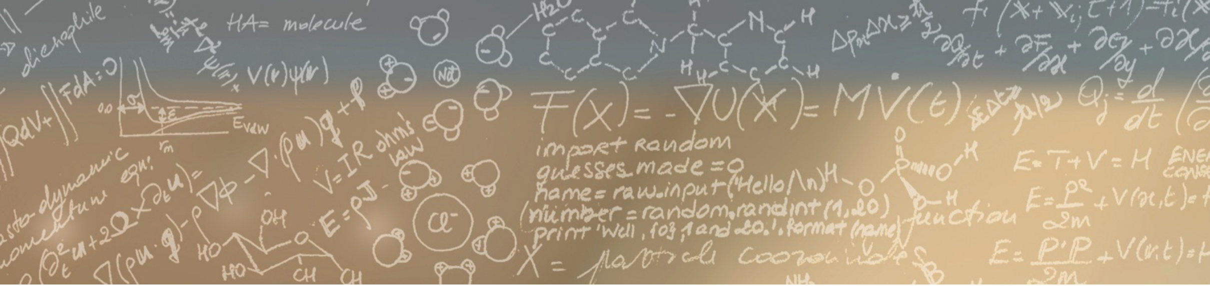
1- Before starting: Create a `.jupyterhub.env` file on `$HOME`

```
module load daint—gpu
module load TensorFlow/1.12.0—CrayGNU—18.08—cuda—9.1—python3
```

2- Launching the Jupyter Notebook server

- https://jupyter.cscs.ch
- Piz Daint node type: GPU
- Queue: not needed for the course
- Training course reservation: **tensor11**
  (Entering a reservation will override the 'Queue' selection)
- Number of nodes: 1 & Job duration: 1 hour
- Start IPyParallel & Dask? No, No

3- Open a Terminal from the dashboard and run `module list`


HA= molecule
import random
guesses_made = 0
name = raw_input('Hello\n')
number = random.randint(1, 20)
print 'Well, {0}, I and 20.'.format(name)
V=IR ohm's law
E=T+V=H

Thank you for your attention!