Centre de Calcul de l'Institut National de Physique Nucléaire et de Physique des Particules

Containers on HPC

October 3rd 2018

PRACE in a few words

Partnership for Advanced Computing in Europe

Mission

- « Enable high-impact scientific discovery and engineering research and development across all disciplines to enhance European competitiveness for the benefit of society »
- « Strengthen the European users of HPC in industry »

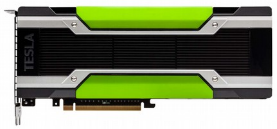
Key figures

- 26 members (one member per Member State of the European Union)
- +500 scientific and industrial projects
- +150 PhD theses, +500 publications, +700 scientific talks supported
- +14 thousand million core hours allocated since 2010

CC-IN2P3 contribution to PRACE

Mission

- Evaluate Tensorflow (Deep Learning framework) by comparing benchmarks with other HPC sites (IDRIS, CINECA...)
- Provide procedures, best practises, advices on deployment and usage of these technologies
- Provide HPC resources

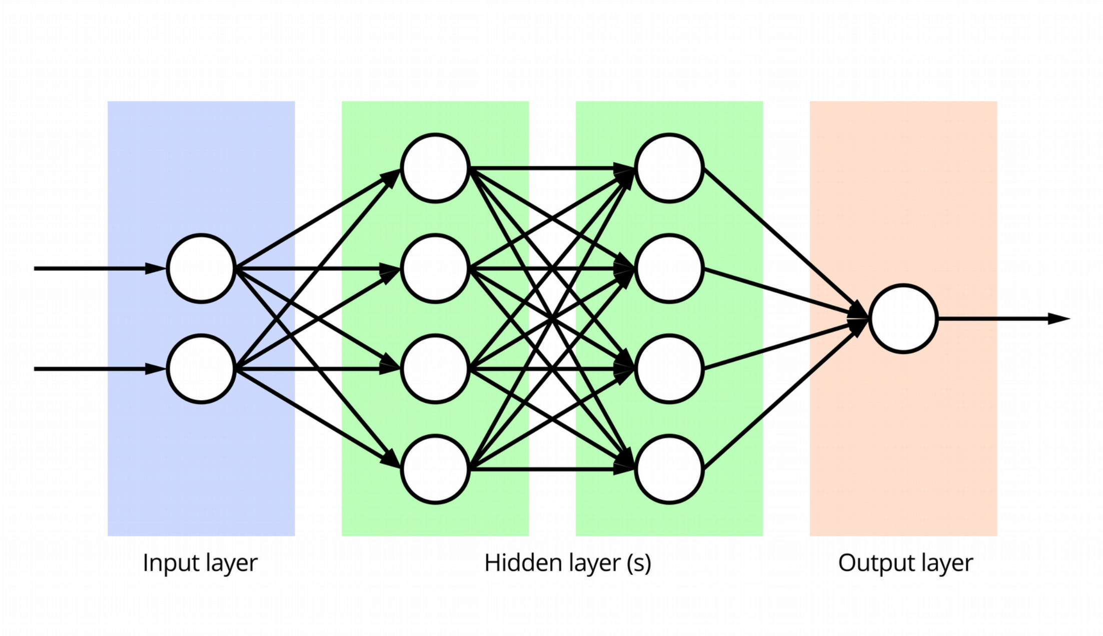


Infrastructure

- 10 workers: Intel(R) Xeon(R) CPU E5-2640 (16 cores) / 128GB RAM
- 40 gpus: Nvidia Tesla K80

What is it?

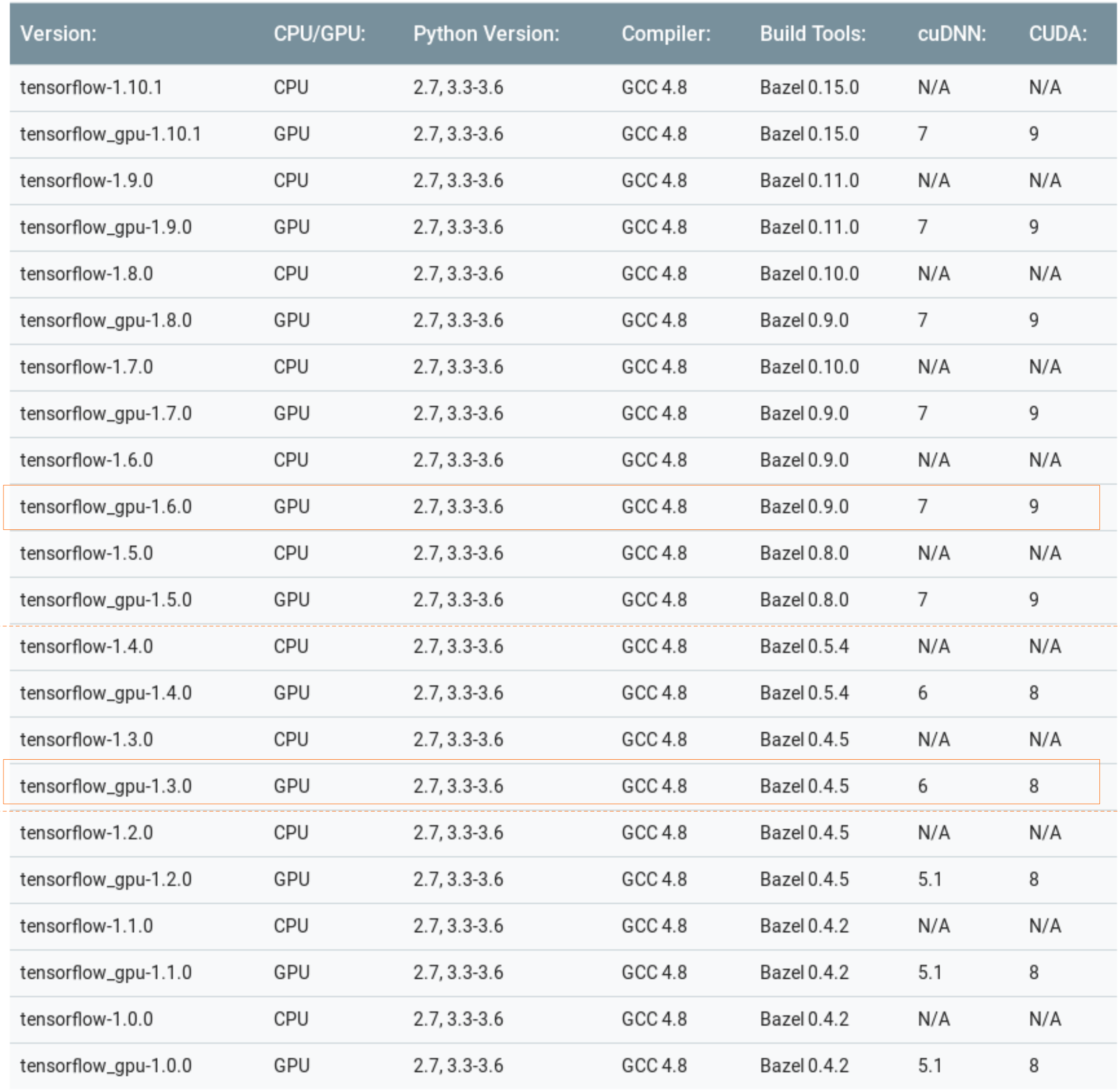
- Tensorflow is a Deep Learning framework
- First developed by Google and now open source
- Provides an abstraction to quickly build models
- Runs on many langages (Python, Java, C++, R, Swift, Go…)
- Same code can be written to run it either on CPUs or GPUs



Tensorflow and Cuda environment

Facts

- Tensorflow and Cuda / Cudnn versions increase rapidly (about a release per trimester)



Version:	CPU/GPU:	Python Version:	Compiler:	Build Tools:	cuDNN:	CUDA:
tensorflow-1.10.1	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.15.0	N/A	N/A
tensorflow_gpu-1.10.1	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.15.0	7	9
tensorflow-1.9.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.11.0	N/A	N/A
tensorflow_gpu-1.9.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.11.0	7	9
tensorflow-1.8.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.10.0	N/A	N/A
tensorflow_gpu-1.8.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	7	9
tensorflow-1.7.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.10.0	N/A	N/A
tensorflow_gpu-1.7.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	7	9
tensorflow-1.6.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	N/A	N/A
tensorflow_gpu-1.6.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	7	9
tensorflow-1.5.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.8.0	N/A	N/A
tensorflow_gpu-1.5.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.8.0	7	9
tensorflow-1.4.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.5.4	N/A	N/A
tensorflow_gpu-1.4.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.5.4	6	8
tensorflow-1.3.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	N/A	N/A
tensorflow_gpu-1.3.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	6	8
tensorflow-1.2.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	N/A	N/A
tensorflow_gpu-1.2.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	5.1	8
tensorflow-1.1.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.2	N/A	N/A
tensorflow_gpu-1.1.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.2	5.1	8
tensorflow-1.0.0	CPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.2	N/A	N/A
tensorflow_gpu-1.0.0	GPU	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.2	5.1	8

Benchmark

- Done with Tensorflow 1.3 (no problem)
- Want now to check again with Tensorflow 1.6 and see differences

Singularity use case : Benchmarks on GPU Farm (PRACE)

Impossible equation

- GPU Farm is set to Cuda 8.0 and Cudnn 6.0
- Tensorflow 1.6 requires Cuda 9.0 and Cudnn 7.0

Solution 1

- Ask SysAdmins to update farm, which is not viable

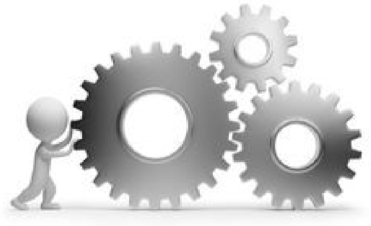
Solution 2

- Use Singularity, becareful of security aspect, and need more maintenance but is pretty agile

Tensorflow & Hardware capabilities

Moreover

- I want to benchmark with a compiled from sources version of Tensorflow to get benefits from all hardware acceleration capabilities



Singularity use case : Benchmarks on GPU Farm (PRACE)

Step 1

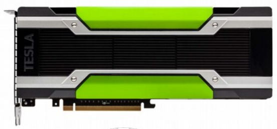
- **Build** a compiler image

- Centos 7
- Cuda 9.0 / Cudnn 7.0 libraries
- Compiler environment (Bazel)
- Python Environment (Miniconda)

Singularity use case : Benchmarks on GPU Farm (PRACE)

Step 2

- **Execute** compiler image from a gpu worker (interactive or qsub)



- Centos 7
- Cuda 9.0 / Cudnn 7.0 libraries
- Compiler environment (Bazel)
- Python Environment (Miniconda)

bash

- Git clone Tensorflow
- Compile

tensorflow-1.6.0-cp36-cp36m-linux_x86_64.whl

Singularity use case : Benchmarks on GPU Farm (PRACE)

Step 3

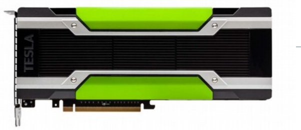
- **Build** a production image

- Centos 7
- Cuda 9.0 / Cudnn 7.0 libraries
- Python Environment (Miniconda)
- Pip install tensorflow-1.6.0-cp36-cp36m-linux_x86_64.whl

Singularity use case : Benchmarks on GPU Farm (PRACE)

Step 4

- **Execute** production image from a gpu worker (interactive or qsub)



- Centos 7
- Cuda 9.0 / Cudnn 7.0 libraries
- Compiler environment (Bazel)
- Python Environment (Miniconda)
- Tensorflow 1.6 (from sources)

bash

- Benchmark

- Speed gain 15-20% between Tensorflow 1.6 and Tensorflow 1.6 (from sources)

Singularity use case : Benchmarks on GPU Farm (PRACE)

Conclusion & Perspectives

- End User Tool
- The new way of sending jobs [my job + my environment]
- Benchmarks showed about same results in terms of performances with or without singularity on the gpu batch farm
- Good for experiences reproducibility
- Singularity Image Catalog: A new service to provide

https://gitlab.in2p3.fr/brigaud/build_tensorflow_from_sources

https://gitlab.in2p3.fr/brigaud/prace-imagenet-benchmark

https://gitlab.in2p3.fr/brigaud/prace-keras-cifar10-benchmark

https://gitlab.in2p3.fr/brigaud/prace-ramp-astro-benchmark

Any question?

Bertrand.rigaud@cc.in2p3.fr

Thanks for your attention.