The PRACE Data Analytics service

Agnès ANSARI

CNRS/IDRIS

Overview

- The Data Analytics service
- Technology used and infrastructure characteristics
- Results:
  - Basic benchmarks
  - CIFAR-10
  - Ramp Astro V1
  - ImageNet
- User working environment
- Integration in the HPC environment
- Advanced features:
  - The dataset download service
  - The Data Analytics GitLab
- Conclusion

The PRACE Data Analytics service

- Goal: to offer a Data Analytics service to PRACE users
  - To investigate the deployment of new Data Analytics technologies on HPC systems
  - To develop prototype solutions by means of pilot scientific use cases to access the functionalities
  - To evaluate how the new prototypes could be adopted as production services in a next phase
  - The PRACE working group involves 6 partners (CNRS/IDRIS, CNRS/IN2P3, CINECA, EPCC, PSNC, KTH)

What will be offered to users:

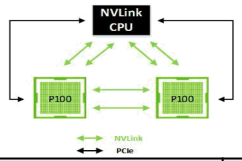
- A set of DL/ML frameworks, tools and libraries which are guaranted to be available on each PRACE Data Analytics system running different architectures (some will be deployed within containers)
- to offer any additional tools or services that facilitate the use of the Deep Learning and Machine Learning services on the PRACE infrastructure and allow users to quickly gain expertise in these fields
  - a dataset download service:
    - *use a large storage space to store large datasets*
  - an environment to gain expertise and to share experience and results:
    - *use of the PRACE shared repository*

Technologies used

- **Deep Learning Frameworks:**
  - Theano (U Montreal) → Tensorflow (Google)
  - Caffe (UC Berkeley) → Caffe2 (Facebook)
  - [Keras : High-level wrapper, python layer on top of Tensorflow (and Theano)] – Easy to build complete solutions, each line of code creates one layer of the network
  - Horovod Distributed deep learning framework for Tensorflow

- **(Distributed Machine learning framework: Spark)**

Infrastructure characteristics



Infrastructure	Type	Architecture /CPU	Accelerators	Librairies/ Applications	LRMS	Environment configuration	OS/Storage
CNRS/ IDRIS	Prototype cluster Ouessant	• OpenPower 8 • 2 Power 8/ node Processors with NVLink bus	• 4 GPUs/node (Nvidia Tesla P100 SXM2 - 16 GB)	• PowerAI framework (TensorFlow, Caffe, Torch, Theano) • Compiled Tensorflow with Keras • Distributed environments: Horovod, DLL, gRPC	Spectrum LSF	• Module • Secured docker installation to run PoweAI jobs • Anaconda or Miniconda	• RHEL 7.3 • IBM Spectrum Scale (GPFS): 128 TB
CNRS/ CC-IN2P3	Production cluster	• Intel x86_64 • 2 Xeon E5 2640 v3/node	• 4 GPUs/node (Nvidia Tesla K80 – 12 GB)	• Compiled TensorFlow with Keras • Distributed environments: Horovod	Univa Grid Engine	• Module • Singularity • Miniconda	• CentOS • GPFS
CINECA	Prototype cluster Davide	• OpenPower 8 • 2 Power 8/ node Processors with NVLink bus	4 GPUs/node (Nvidia Tesla P100 SXM2 - 16 GB)	• Compiled Tensorflow with Keras • Distributed environments: Horovod, gRPC	SLURM	• Module	• CentOS 7.5 • NFS 697 TB
EPCC	Production cluster Urika-GX	• Intel x86_64 • 2 Xeon E5-2695/node 256 GB	None	• Intel Tensorflow • BVLC Caffe • Intel Caffe	MESOS	• Anaconda • Virtual conda environments	
EPCC	Cirrus	• Intel x86_64 • 2 Xeon E5-2695/node 256 GB	None	• Intel Tensorflow • BVLC Caffe • Intel Caffe	PBS		
CINECA	Marconi A2	• Intel x86_64 • Xeon Phi 7250	None	• Compiled TensorFlow with Keras	SLURM	• Module	• CentOS 7.2 • GPFS 10 PB
CINECA	Galileo	• Intel x86_64 • 2 Xeon E5-2630/node	2 GPUs/node (Nvidia K80)	• Compiled Tensorflow with Keras • Distributed environments: Horovod, gRPC	SLURM	• Module	• CentOS 7.0 • GPFS 5PB

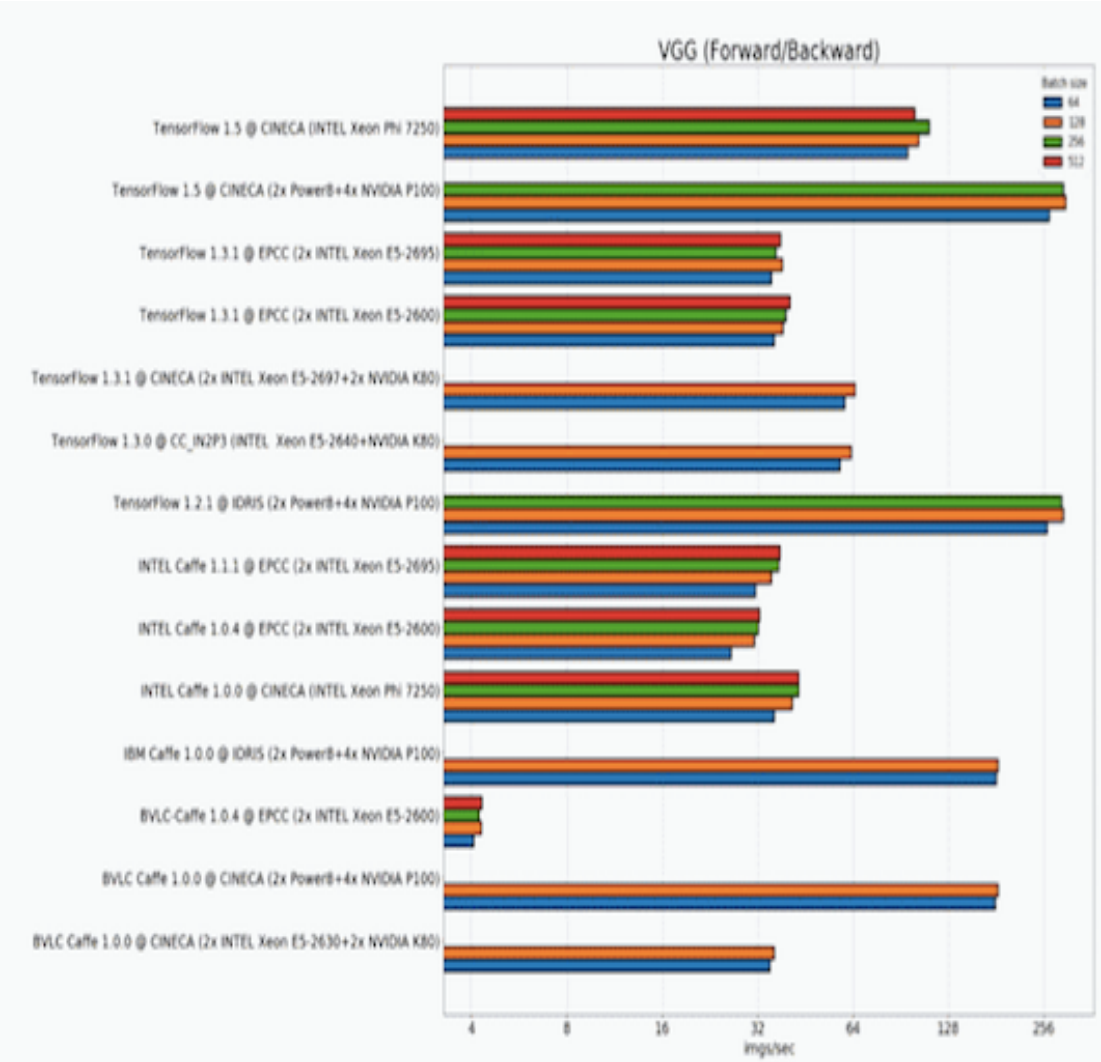
Basic benchmarks

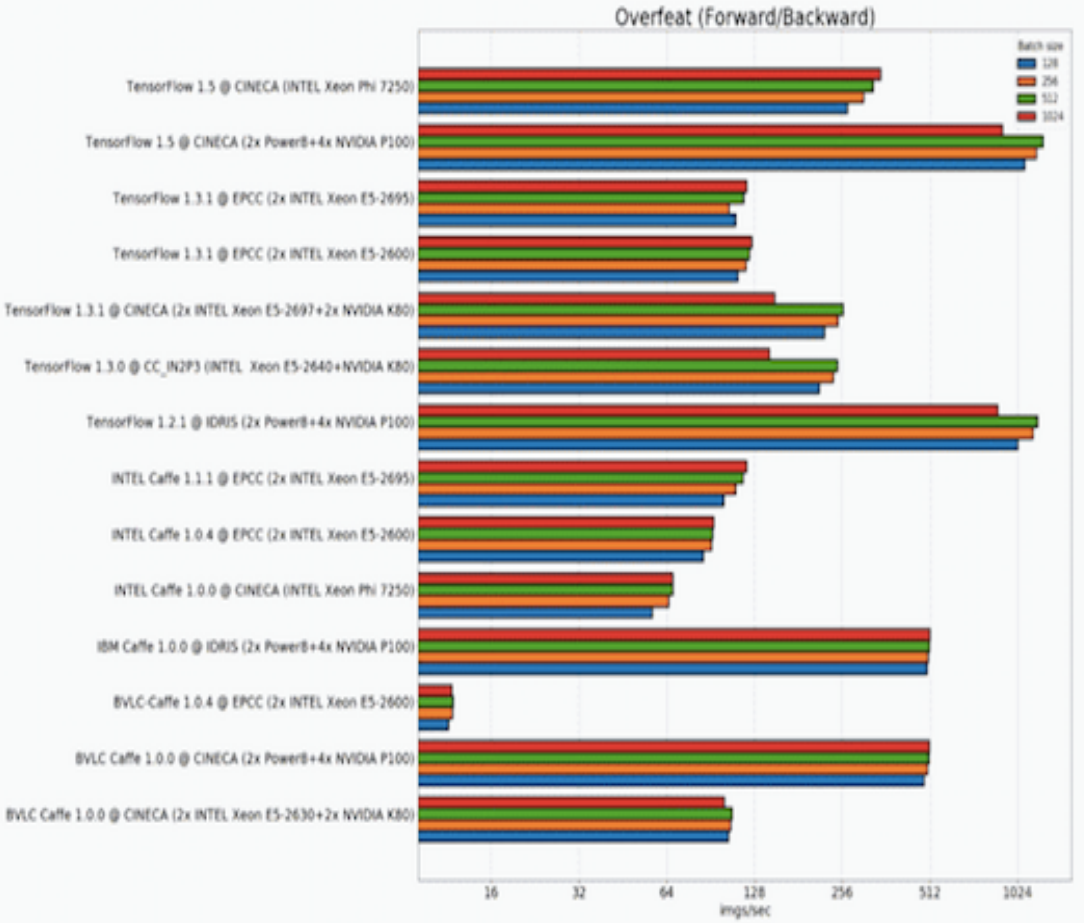
- Based on the Convnet benchmarks project
- Simplified approach with only the convolutional fully connected layers
- No dataset loading, data representing a image batch is created in memory
- The throughput is tracked in terms of images per second for both forward (inferencing) and forward-backward (learning) steps

- TensorFlow or Caffe
- Model trained (AlexNet, GoogLeNet, Overfeat and VGG11).
- Mode (Backward/Forward and Forward)
- With a fixed number of iterations (100), on one GPU only, and for a varying set of batch size

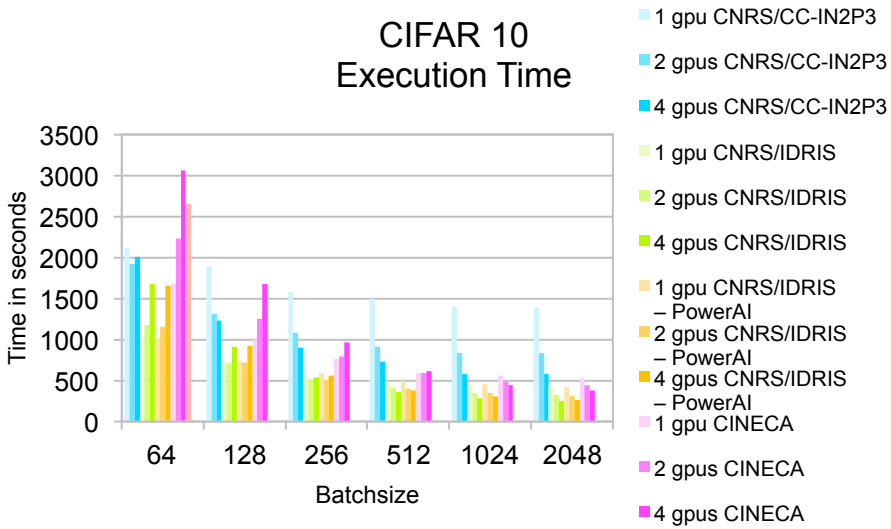
Basic Benchmarks

IDRIS	Power 8 + Nvidia P100	PowerAI - Tensorflow 1.2.1/ IBM Caffe 1.0.0
CINECA 1	Xeon Phi 7250	Tensorflow 1.5/Intel Caffe 1.0.0
CINECA 2	Power 8 + Nvidia P100	Tensorflow 1.5/IBM Caffe 1.0.0/BVLC Caffe 1.0.0
CINECA 3	Xeon E5-2697 + Nvidia K80	Tensorflow 1.3.1/BVLC Caffe 1.0.0
CC-IN2P3	Xeon E5-2640 + Nvidia K80	Tensorflow 1.3.0
EPCC	Xeon E5-2695	Tensorflow 1.3.1/ Intel Caffe
BS	Batch size	





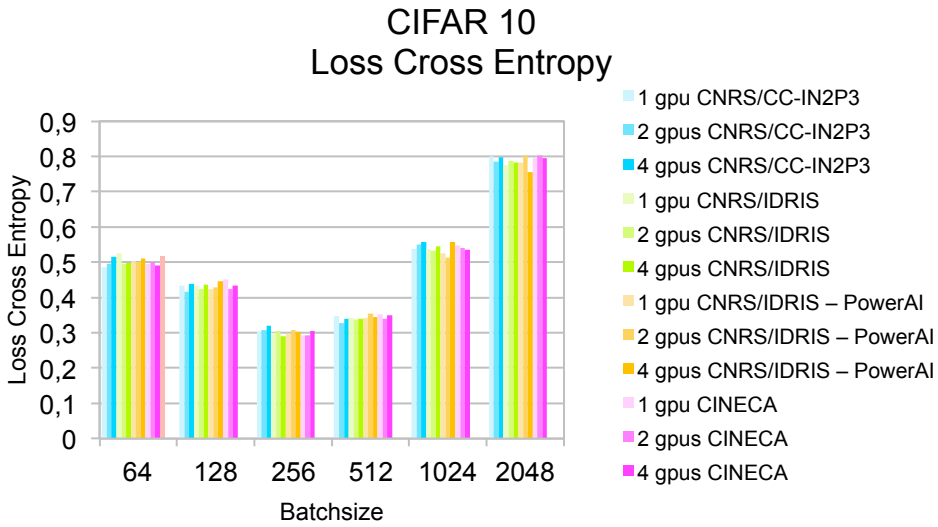
CIFAR-10 Benchmark

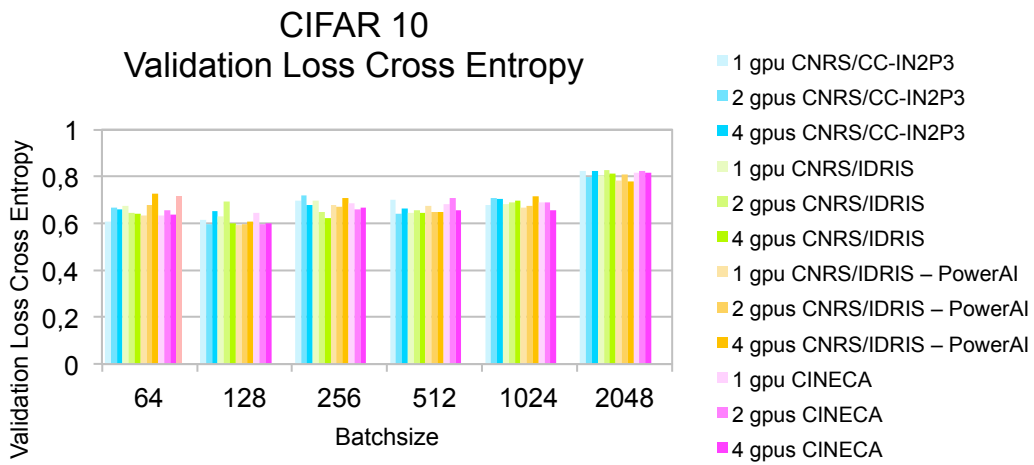


Site/Cluster Name/GPU	Frameworks	Librairies
CINECA – Davide – Nvidia P100	TensorFlow 1.8	Cuda 9.2, Cudnn 7.1, Keras
CNRS/CC_IN2P3 Nvidia K80	TensorFlow 1.8	Cuda 9.2, Cudnn 7.0.5, Keras 2.2.0
CNRS/IDRIS – Ouessant Nvidia P100	PowerAI/TensorFlow 1.8 TensorFlow 1.8	Cuda 8.0, Cudnn 6, Keras 2.1.2 Cuda 9.2, Cudnn 7, Keras 2.2.0

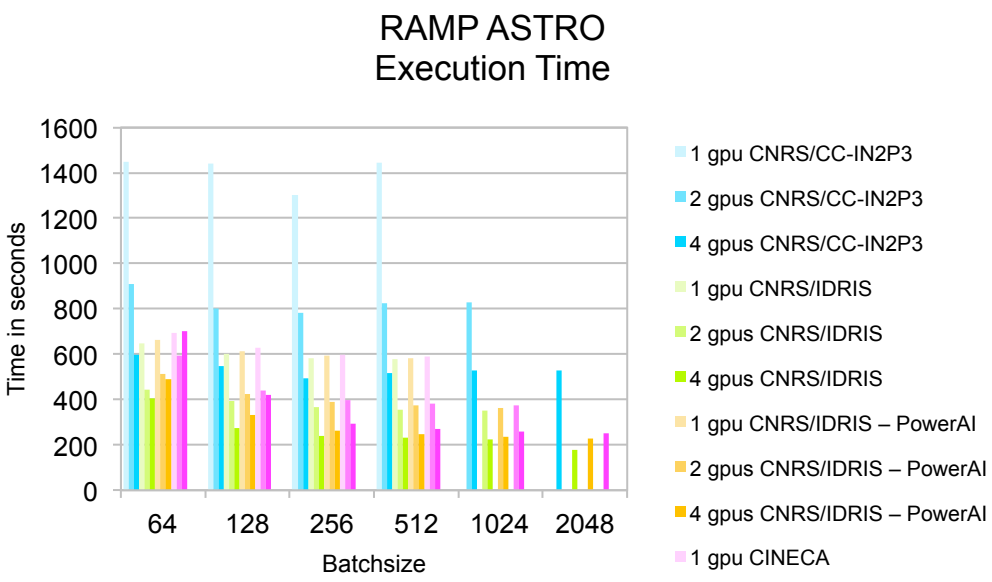
Model	Simple CNN, 12 layers
Batch size	64, 128, 256, 521, 1024, 2048

Dataset details	CIFAR-10
Elements	32x32 RGB pictures
Total number of elements	60 000
Number of Training elements	50 000
Number of Validation elements	10 000
Categories	10
Dataset size	163 MiB



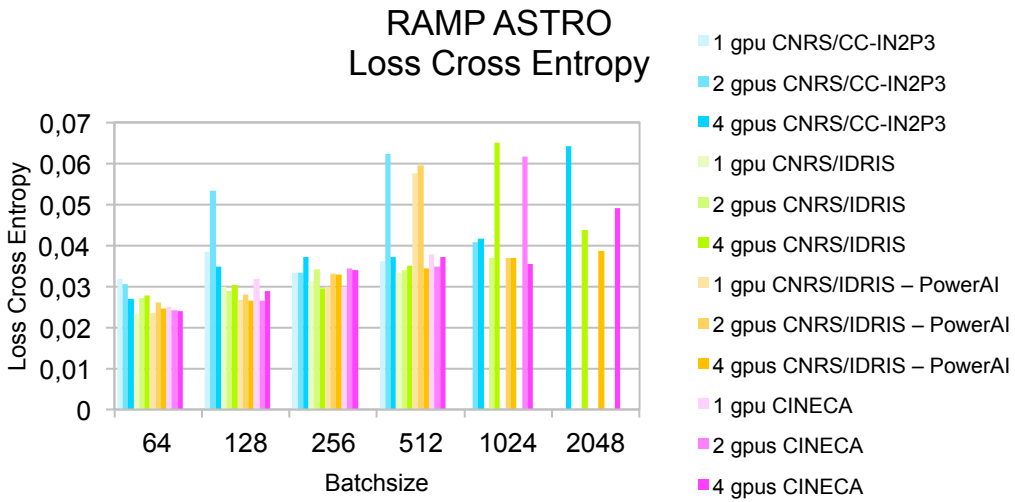


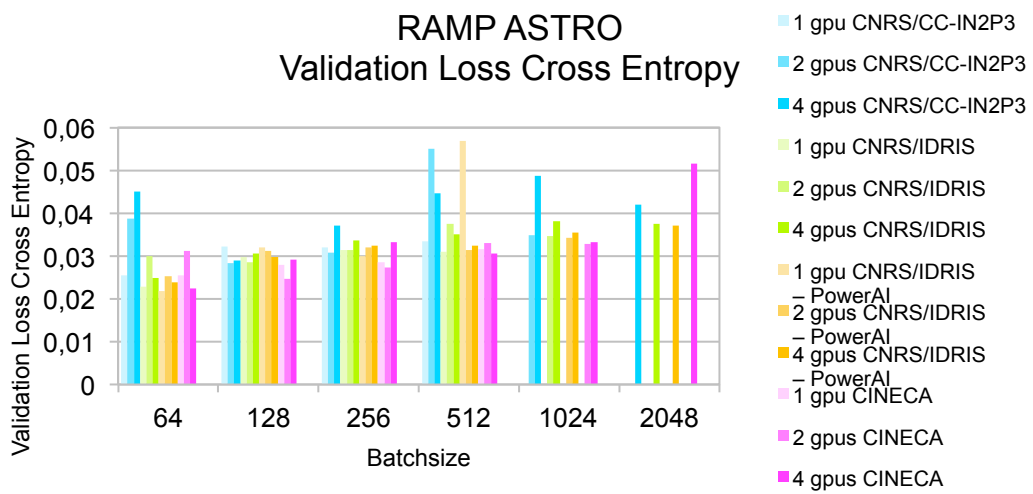
Ramp Astro Benchmark V1



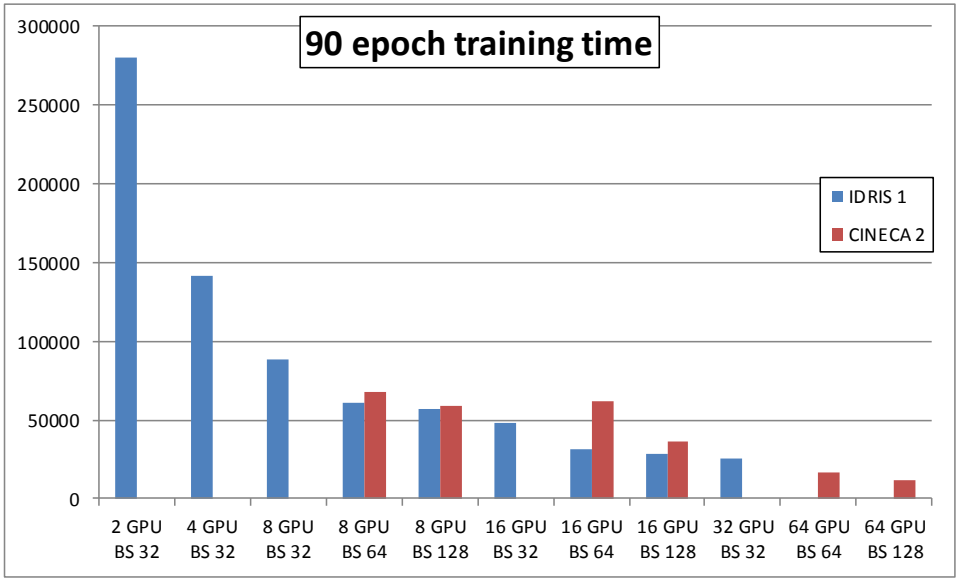
Site/Cluster Name/GPU	Frameworks	Librairies
CINECA – Davide – Nvidia P100	TensorFlow 1.10	Cuda 9.2, Cudnn 7.1, Keras
CNRS/CC_IN2P3 Nvidia K80	TensorFlow 1.8	Cuda 9.2, Cudnn 7.0.5, Keras 2.2.0
CNRS/IDRIS – Ouessant Nvidia P100	PowerAI/TensorFlow 1.10 TensorFlow 1.8	Cuda 8.0, Cudnn 6, Keras 2.1.2 Cuda 9.2, Cudnn 7, Keras 2.2.0

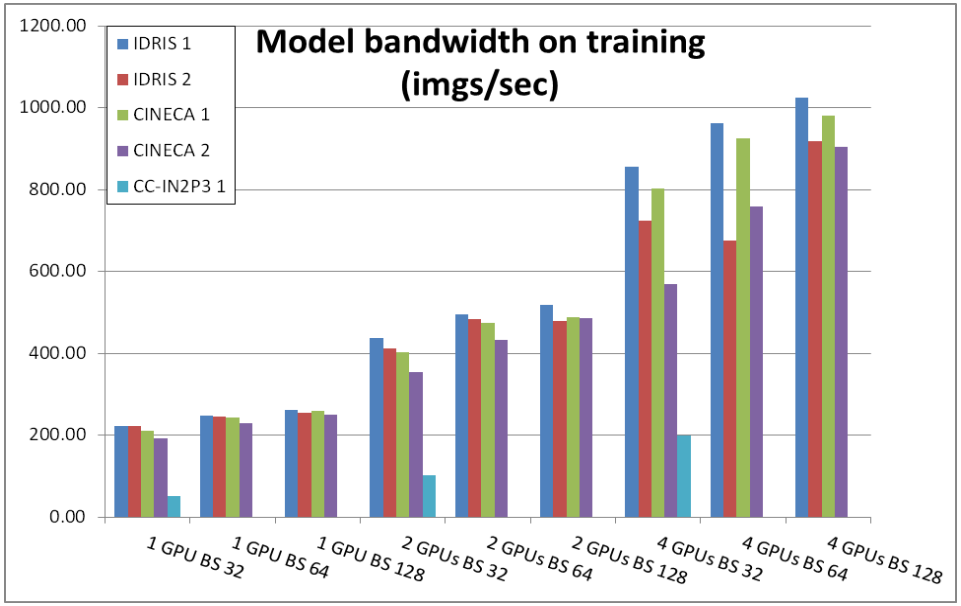
Dataset	- 2 GB - 12 000 images for the training, - 4 000 for the validation (during training) - 4 000 images for the final evaluation of the model
Model	UNet
Batch size	64, 128, 256, 521, 1024, 2048





ImageNet Benchmark (1)

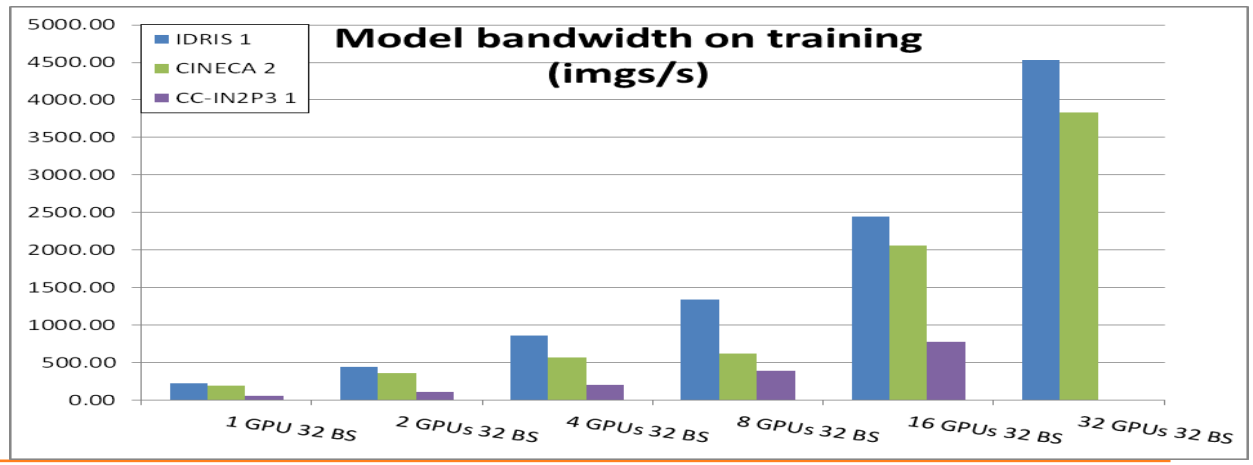




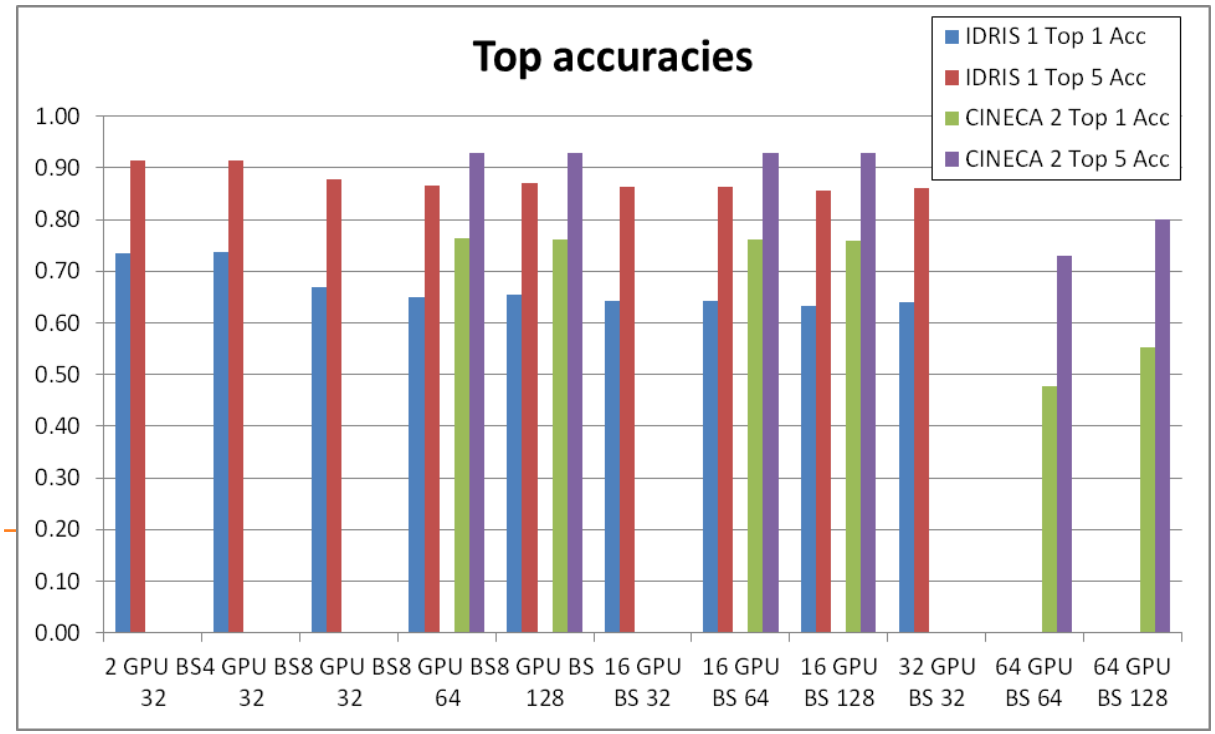
Site/Cluster Name	Frameworks	Librairies
CINECA – Davide NVIDIA P100	1. ***TensorFlow*** 1.8, Horovod 0.14.1 2. ***TensorFlow*** 1.10, Horovod 0.13.11	CUDA 9.0, CuDNN 7.0
CNRS/CC_IN2P3 NIVIDIA K80	***TensorFlow*** 1.8/Singularity	CUDA 9.0, CuDNN 7.0
CNRS/IDRIS – Ouessant NVIDIA P100	1. ***TensorFlow*** 1.8, Horovod 0.13.3, (Caffe 1.0) 2. ***PowerAI/TensorFlow*** 1.8, (Power/AI Caffe 1.0)	CUDA 9.2, CuDNN 7.1

Model	***ResNet50***, (VGG-19)
Batch size	32,64,128

Dataset details	ImageNet: ILSVRC 2012
Elements	RGB annotated pictures 400x350 on average
Total number of elements	1 250 000
Number of Training elements	1 200 000
Number of Validation elements	50 000
Categories	1000
Dataset size	138 GB (Training) + 6.3 GB (Validation) + 13 GB (Test set)



ImageNet Benchmark (2)



Tensorflow [ResNet50]: Top validation accuracies (90 epochs) using basic hyper parameters (no tuning performed)

➢ For 1epoch, the Caffe [VGG-19] performances (training time and images/second) appear to be:
- between 4 and 5 times worse for 1 and 2 GPU
- between 10 and 12 times worse for 4 GPUs

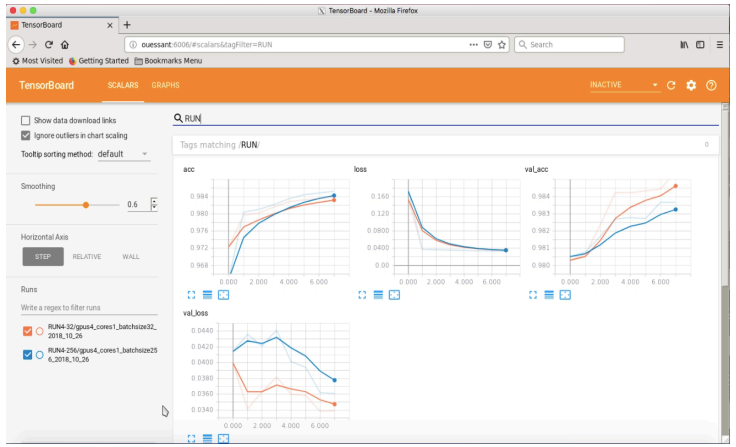
than the Tensorflow [ResNet50] ones

➢The increase in the number of GPUs and a larger batch size have an impact on the model accuracy, lowering the accuracy and incrementing the validation error

➢A tradeoff must be achieved between accuracy and performance in the training process

User working environment

- To guarantee the effectiveness of the training:
  - Monitor the training execution: nvidia-smi, htop, nmon
  - Check the training progress and visualization:
    - *tensorboard to check statistics over the time or compare multiple executions*



  - Debug:
    - *console, batch output,*
    - *Horovod profiling tool Timelines*
  - Checkpoint the training process:
    - *feature from Tensorflow or Caffe (model architecture, weights, configuration)*
    - *batch scheduler level*

Tensorflow integration in the HPC environment (1)

- DL frameworks evolves quickly, with multiples releases
- Quick loss of compatibility between the various components which makes the installations complicated
- Solutions used in the PRACE Data Analytics working group to offer the required flexibility:
  - Conda virtual environments (users shared environments at IDRIS)
  - Containers
    - *Singularity*
    - *LSF/Docker (secure configuration at IDRIS)*

Tensorflow integration in the HPC environment (2)

```
# Simple Deep Learning job skeleton without container

# Batch scheduler resources specification

# Load the user environment
ml tensorflow/py3/1.10

#Load the shared virtual environment
source activate tensorflow 1.10-py3

# Run the DL application
python ./myNN.py
```

```
# Simple Deep Learning job skeleton withcontainer

# Batch scheduler resources specification

# Load the user environment
module load singularity


# Run the DL application using Singularity (or Docker)
singularity exec tflow.img myTraining.sg
```

```
# Simple Deep Learning job skeleton with LSF/
DOCKER
#  Before submitting the job:
# ml powerai/py3/1.5.2; source activate powerai-py3

# Batch scheduler  resources specification
# BSUB –app docker
…
source /opt/DL/tensorflow/bin/tensorflow–activate

# Run the DL application
python ./myNN.py
```

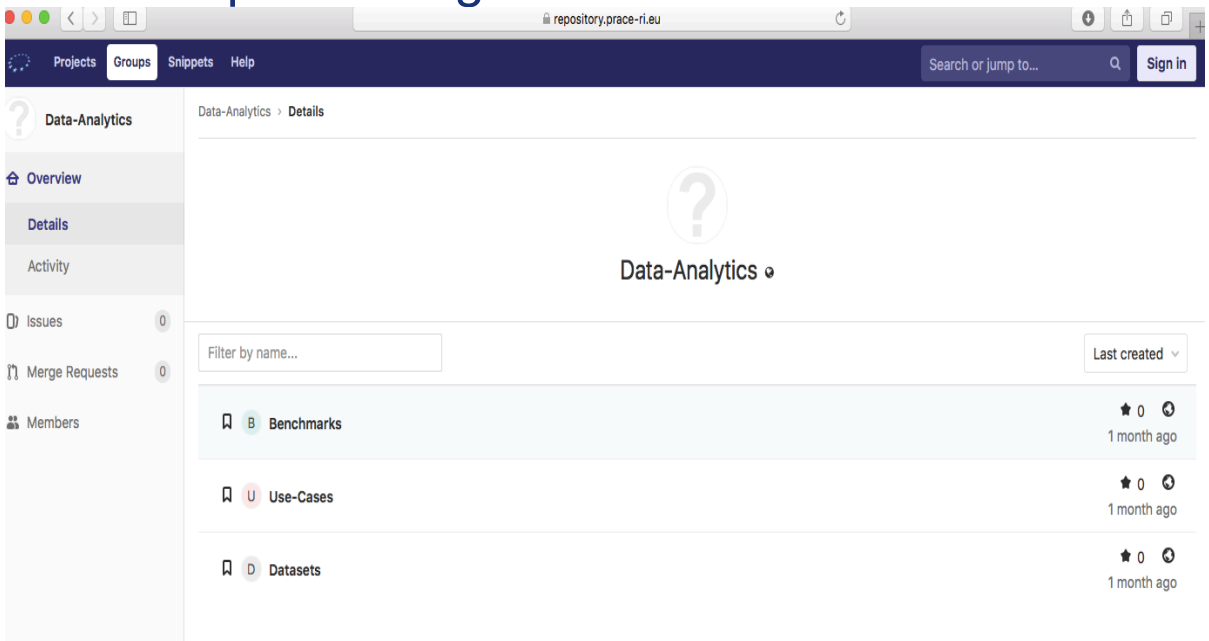
Spectrum LSF configuration file
Application profile definition

```
# PowerAI

Begin Application

CONTAINER = docker[image(registryhost:port/
powerai) options( …  )]

NAME = docker

End Application
```

Advanced features (1) – The dataset download service

- to offer users easy access to datasets that are commonly used in the AI domain or to any other specific datasets
- to share these datasets in a dedicated storage space somewhere in the PRACE infrastructure in order to benefit from the fast, reliable and secure PRACE VPN network and avoid tedious download from internet
- Implementation:
  - service provided by CINECA, one TB of sorage disk
  - implemented through iRODS
  - currently used by EUDAT B2SAFE
  - provides both dataset download and upload capabilities to the PRACE users

Advanced features (2)
PRACE GitLab: The Data Analytics project group
Environment to gain expertise and to share experience
Description using metadata



List of available benchmarks

CIFAR10 benchmark:

- Authors: provided by Bertrand Rigaud - bertrand.rigaud@cc.in2p3.fr
- Description: See README.md
- Type: script
- Language: python 3
- Environment: python 3.6+, tensorflow, Keras
- Tag:
- Output: data files and visualization: - logs files (can be viewed via tensorboard) - neural network model
- url: https://repository.prace-ri.eu/git/Data-Analytics/Benchmarks/tree/master/cifar10_benchmark

List of available datasets

CIFAR-10: Computer-vision images dataset used for object recognition

- Description: The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.
- Size: 163 MB (python version)
- File type: compressed archive
- License: If you're going to use this dataset, please cite the tech report: Learning Multiple Layers of Features from Tiny Images, Alex Krizhevsky, 2009.
- Tag:
- Prace dataset location:
- url: https://www.cs.toronto.edu/~kriz/cifar.html
- Dataset download url:https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz, md5sum: c58f30108f718f92721af3b95e74349a
- Prace Download script: https://repository.prace-ri.eu/git/Data-Analytics/Datasets/blob/master/cifar10-download.sh

==========

ImageNet: Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)

- Description: Training data 1,281,167 224x224 colour images in 1000 synsets. Validation data 50 images/synset. Test data 100 images/synset.
- Size:

  Training images (Task 1 & 2). 138GB. MD5: 1d675b47d978889d74fa0da5fadfb00e Training images (Task 3). 728MB. MD5: ccaf1013018ac1037801578038d370da Validation images (all tasks). 6.3GB. MD5: 29b22e2961454d5413ddabcf34fc5622 Test images (all tasks). 13GB. MD5: fe64ceb247e473635708aed23ab6d839

- File type: archive file
- License:

Conclusion

- The PRACE data analytics service is based on prototype services
- Prototype services include: Tensorflow, Caffe, Keras, Horovod
- Tensorflow
  - Remains the major player with a fast community growth
  - Provides performance very far from those of caffe
- Results:
  - The performances depend mostly on the hardware
  - The GPU NVIDIA P100 outperforms the GPU NVIDIA K80
  - The increase in the number of GPUs and a larger batch size have an impact on the model accuracy, lowering the accuracy and incrementing the validation error
  - A tradeoff must be achieved between accuracy and performance in the training process
- Check the effectiveness of the training:
  - Execution monitoring: nvidia-smi, nmon, htop
  - Training progress checking and visualization: Tensorboard
  - Debugging: Horovod TimeLine (other mpi debugging tools)
  - Checkpointing: Tensorflow/Caffe feature, batch scheduler
- Use of advanced features to help users to perform their Data Analytics tasks
  - The dataset download service
  - The PRACE Data Analytics GitLab project
- Still a fast pace technological evolution

QUESTIONS?

www.prace-ri.eu