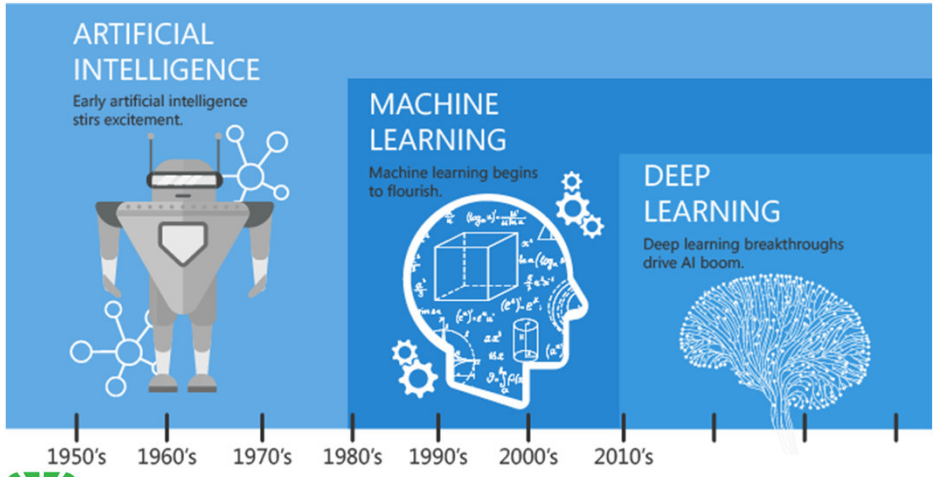
How to run AI programs in Graham

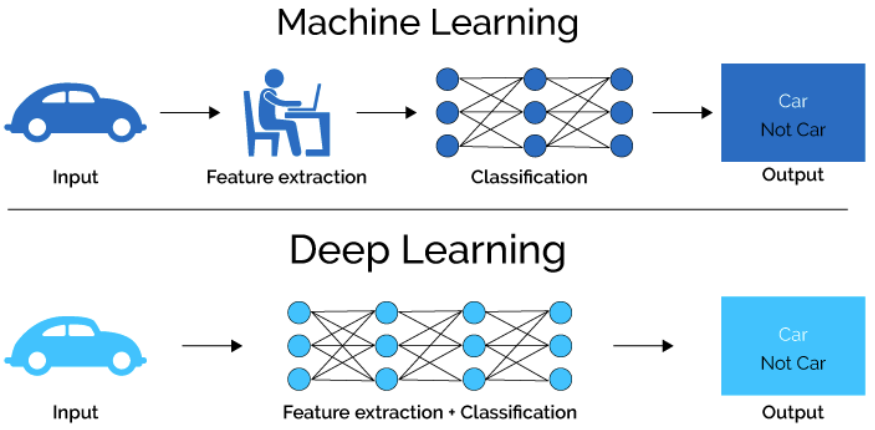
Isaac Ye, HPTC @ York University

Isaac@sharcnet.ca

AI / Deep Learning

Deep learning (DL) is a class of machine learning algorithms in which multiple layers of nonlinear processing units are used for feature extraction and transformation, with each successive layer taking the output from the previous layer as input.



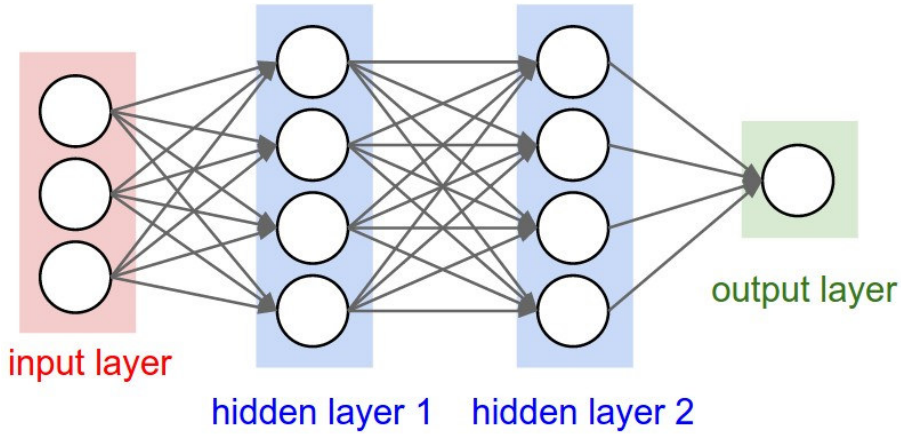


compute · calcul ONTARIO

compute canada | calcul canada

Deep Neural Network (DNN)

"A family of parametric, non-linear and hierarchical representation learning functions, which are massively optimized with stochastic gradient descent to encode domain knowledge, i.e. domain invariances, stationarity." -- Efstratios Gavves

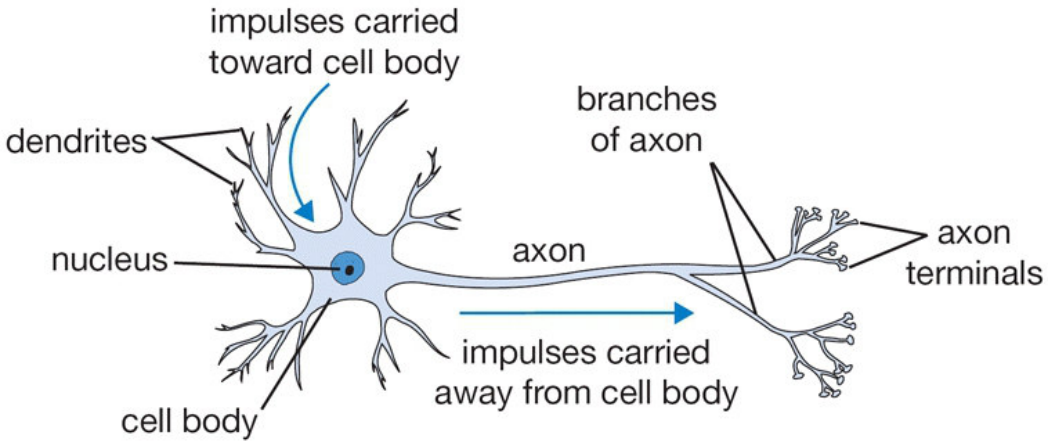


Example of a 3-layer Deep Neural Network (DNN)

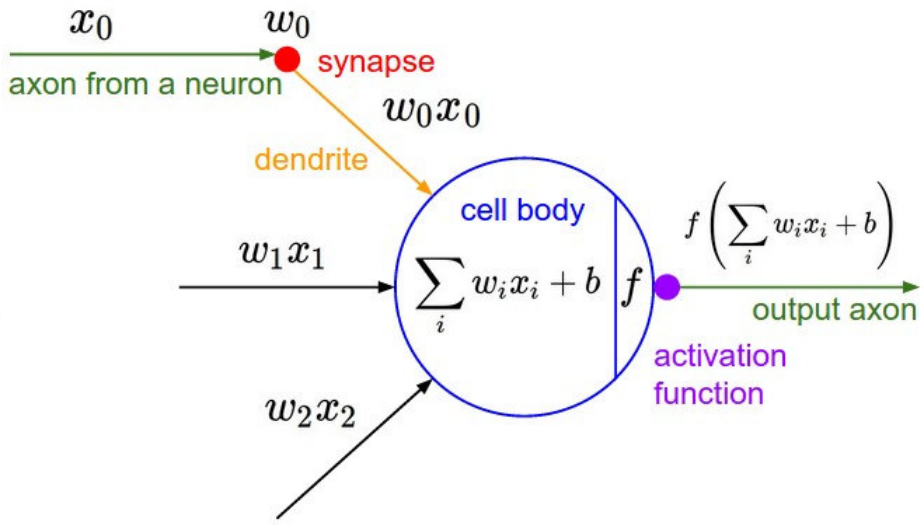
http://cs231n.github.io/neural-networks-1/

Neural Network

http://cs231n.github.io/neural-networks-1/

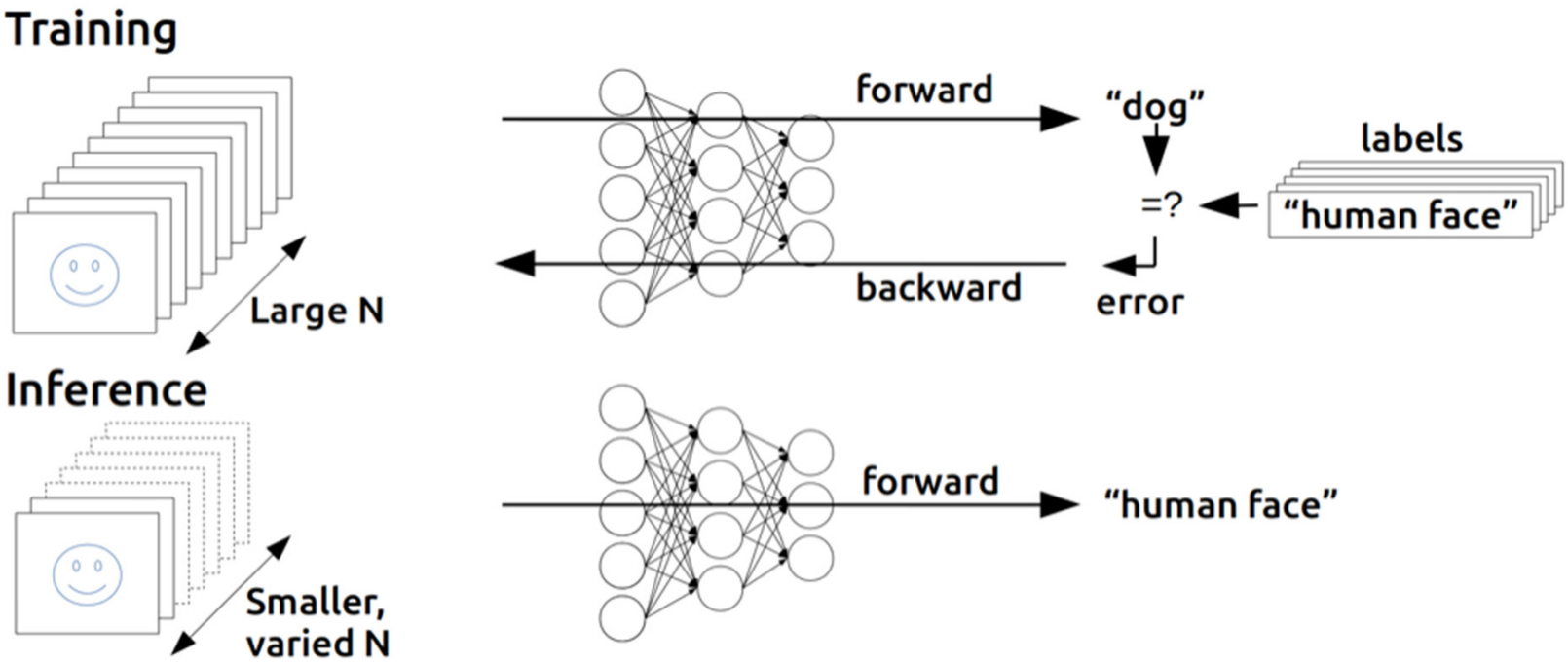


Biological neuron



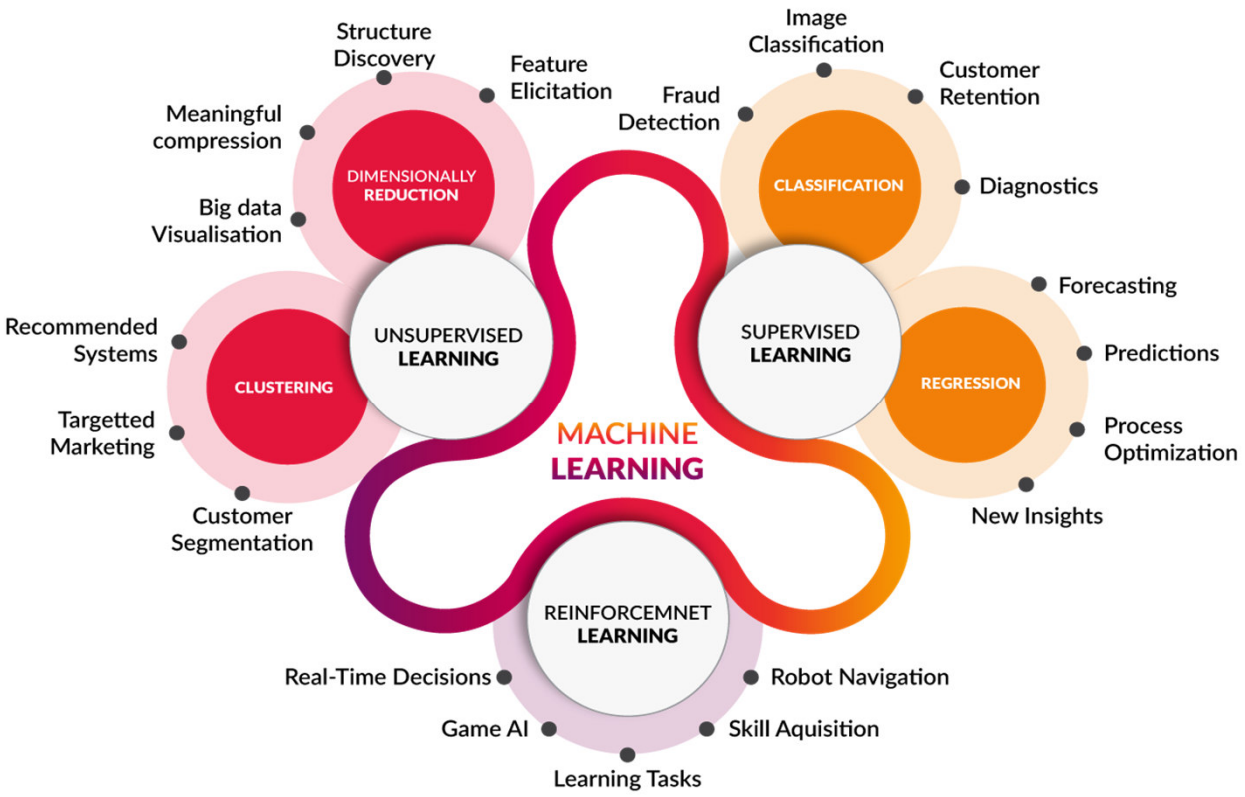
Mathematical model

DL: Training / Inference



https://devblogs.nvidia.com/inference-next-step-gpu-accelerated-deep-learning/

What can we do with AI?



http://www.cognub.com/index.php/cognitive-platform/

Landscape of Science problems

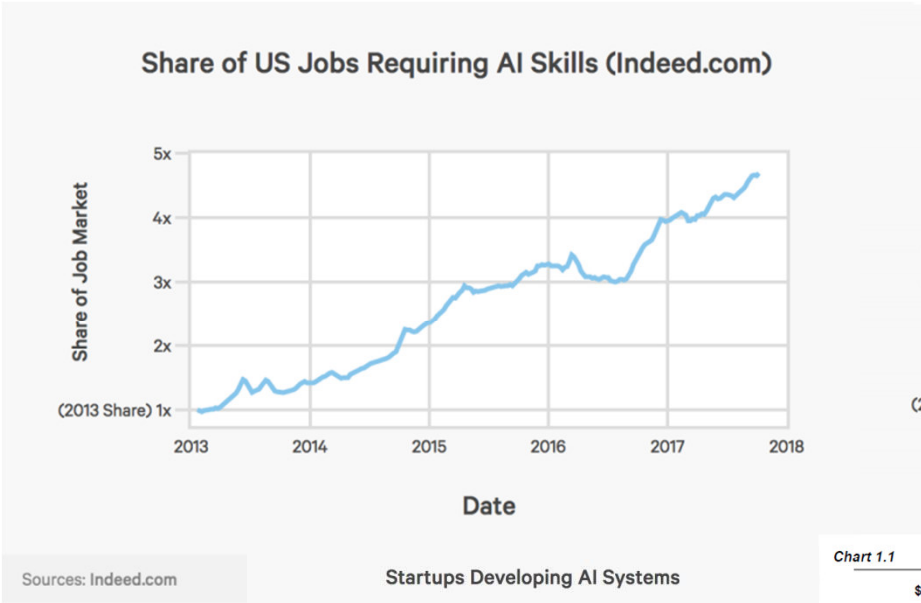
	High Energy Physics			Biological and Environment		Basic Energy		Nuclear Physics	Fusion Energy
	Astronomy	Cosmology	Particle Physics	Climate	Genomics	Light Sources	Materials	Heavy Ion Colliders	Plasma Physics
Classification	X		X	X	X	X	X	X	X
Regression	X	X	X	X	X	X	X	X	X
Clustering	X	X	X	X	X	X	X	X	X
Dimensionality Reduction				X				X	
Surrogate Models	X	X	X	X			X	X	X
Design of Experiments		X		X		X	X		X

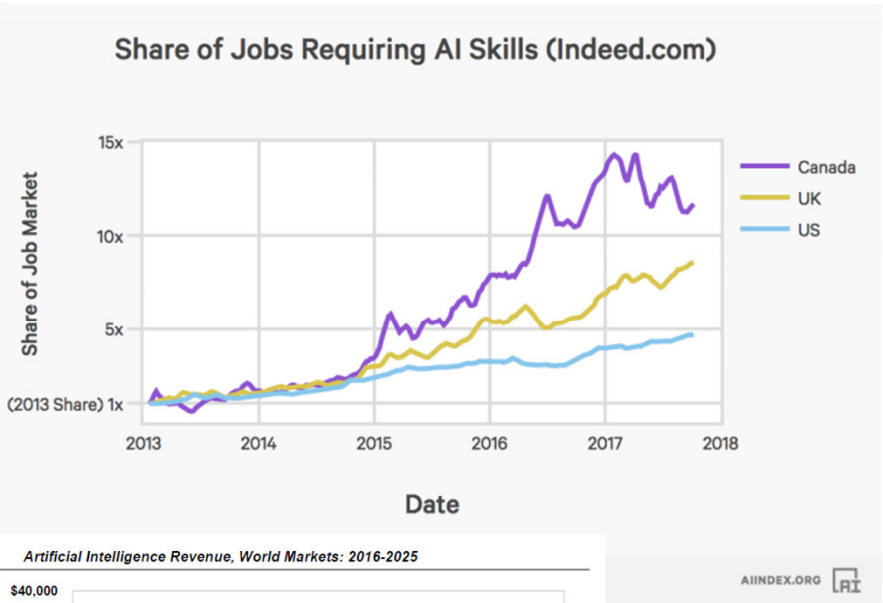
Analytics

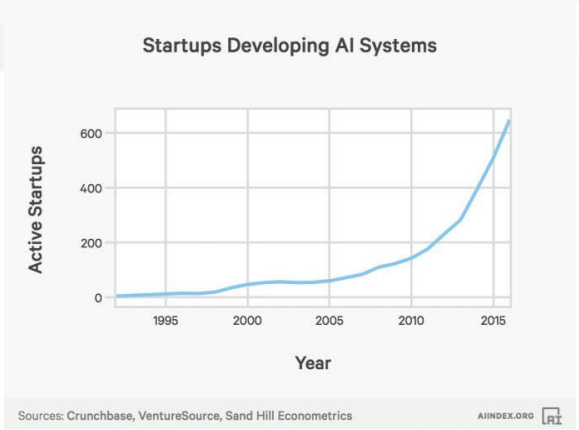
Simulations

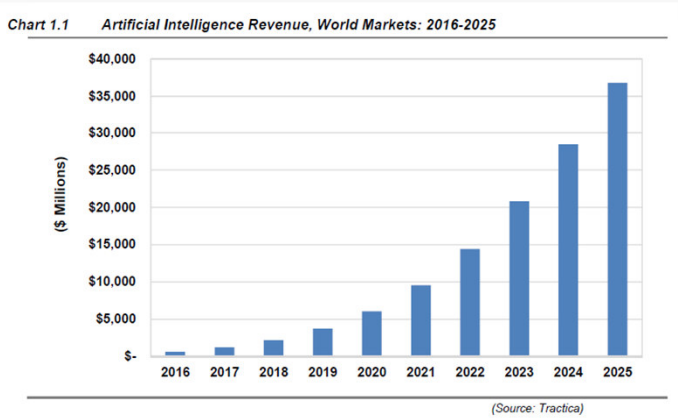
Control

DL use cases and growth trends







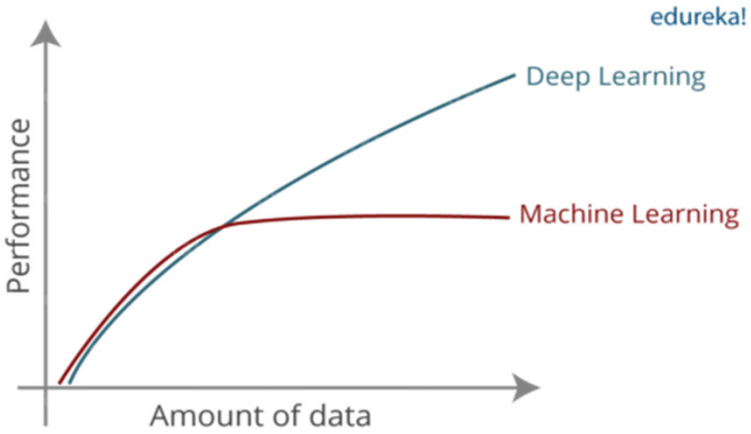


compute · calcul ONTARIO

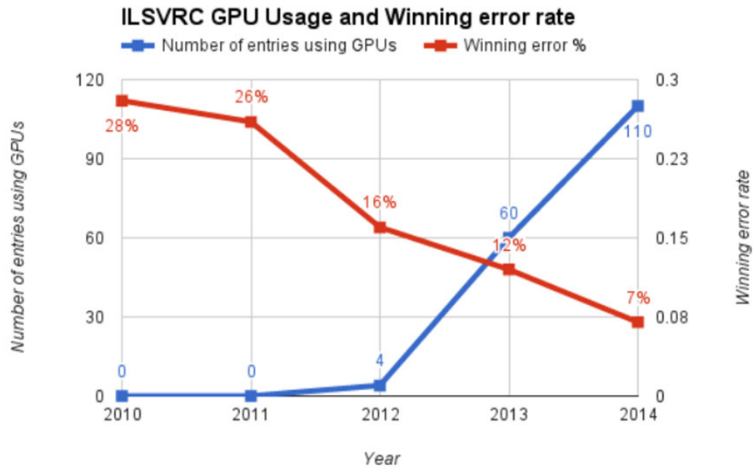
compute canada | calcul canada

Why now?

1) Data: large curated datasets



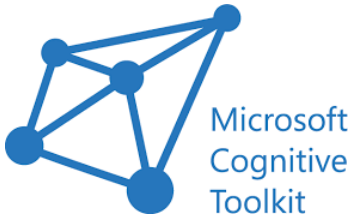
2) GPUs: linear algebra accelerators

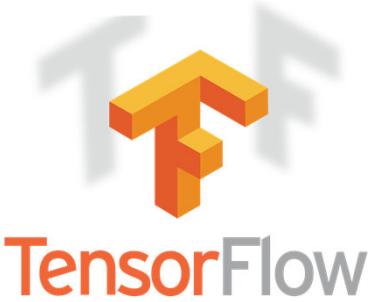


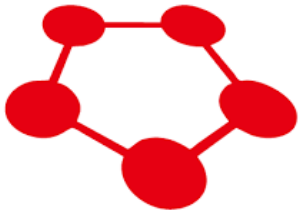
https://devblogs.nvidia.com/nvidia-ibm-cloud-support-imagenet-large-scale-visual-recognition-challenge/

3) Algorithmic advances: optimizers, regularization, normalization … etc.

DL frameworks



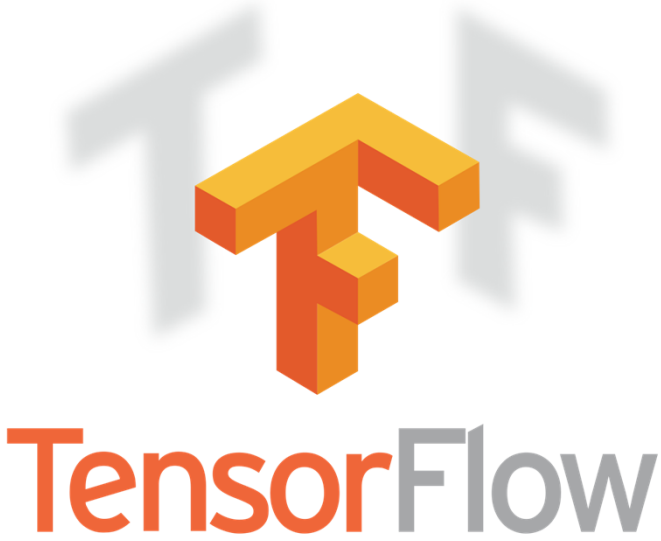




TensorFlow

- The most widely used framework open-sourced by Google
- Replaced Google's DistBelief framework
- Runs on almost all architectures (CPU/GPU/TPU/etc)
- Define-and-Run type for neural networks
- Version 2.0 has Define-by-Run component(Eager execution)
- https://github.com/tensorflow/tensorflow

```
[isaac@cedar1 ~]$ avail_wheels "tensorflow*"
name                    version    build    python    arch
----------------------  ---------  -------  --------  -------
tensorflow_cpu          2.1.0              cp37      generic
tensorflow_cpu          2.1.0              cp36      generic
tensorflow_cpu          2.1.0              cp35      generic
tensorflow_estimator    2.1.0              py2.py3   generic
tensorflow_gpu          2.1.0              cp37      generic
tensorflow_gpu          2.1.0              cp36      generic
tensorflow_gpu          2.1.0              cp35      generic
tensorflow_tensorboard  1.5.1              py3       generic
tensorflow_tensorboard  1.5.1              py2       generic
[isaac@cedar1 ~]$
```



PyTorch

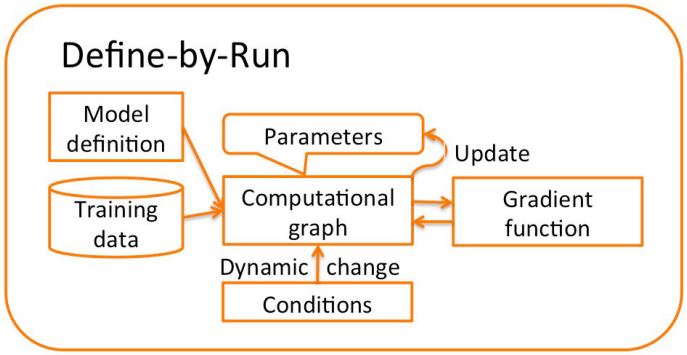
- Rapidly growing in research community for deep learning framework developed by Facebook
- A Python adaptation of Torch
- Caffe2 has been merged to PyTorch
- Define-by-Run type for neural networks
- Ease of expression and use
- https://github.com/pytorch/pytorch

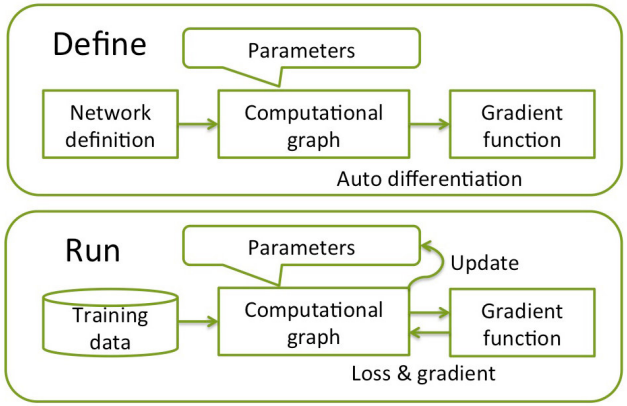

```
[isaac@cedar1 ~]$ avail_wheels "torch_*"
name              version    build    python    arch
----------------  ---------  -------  --------  -------
torch_cluster     1.4.5               cp37      generic
torch_cluster     1.4.5               cp36      generic
torch_cluster     1.4.5               cp35      generic
torch_cpu         1.0.0               cp37      avx2
torch_cpu         1.0.0               cp36      avx2
torch_cpu         1.0.0               cp35      avx2
torch_cpu         1.0.0               cp27      avx2
torch_geometric   1.4.2               py3       generic
torch_gpu         1.0.0               cp37      avx2
torch_gpu         1.0.0               cp36      avx2
torch_gpu         1.0.0               cp35      avx2
torch_gpu         1.0.0               cp27      avx2
```

PyTorch

TensorFlow

Pros	Easy to use (Python support) Intuitive Dynamic graphs Research community prefers	Large community Heterogeneous architecture TF 2.0: Eager execution(Define-by-Run) Tensorboard (visualizing), Keras
Cons	Small community Less additional tools	Verbose Static graphs

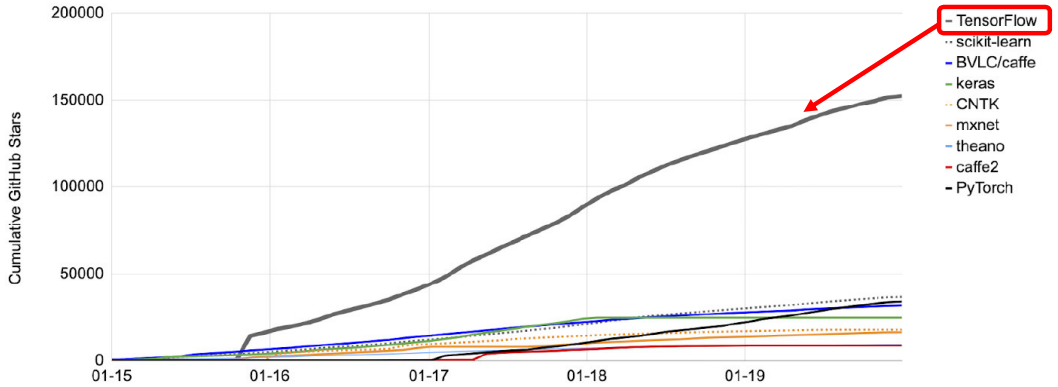




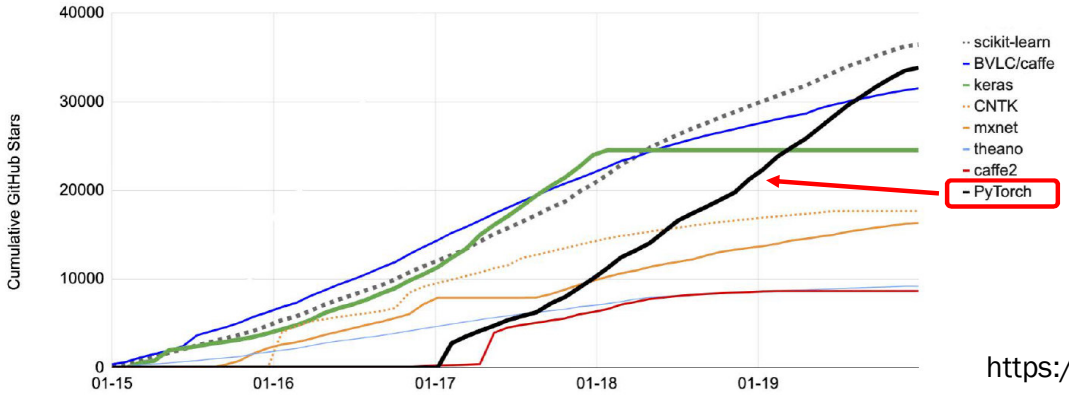
https://www.oreilly.com/content/complex-neural-networks-made-easy-by-chainer/

DL frameworks trend

Cumulative GitHub stars by AI library (2015—2019)
Source: Github, 2019.

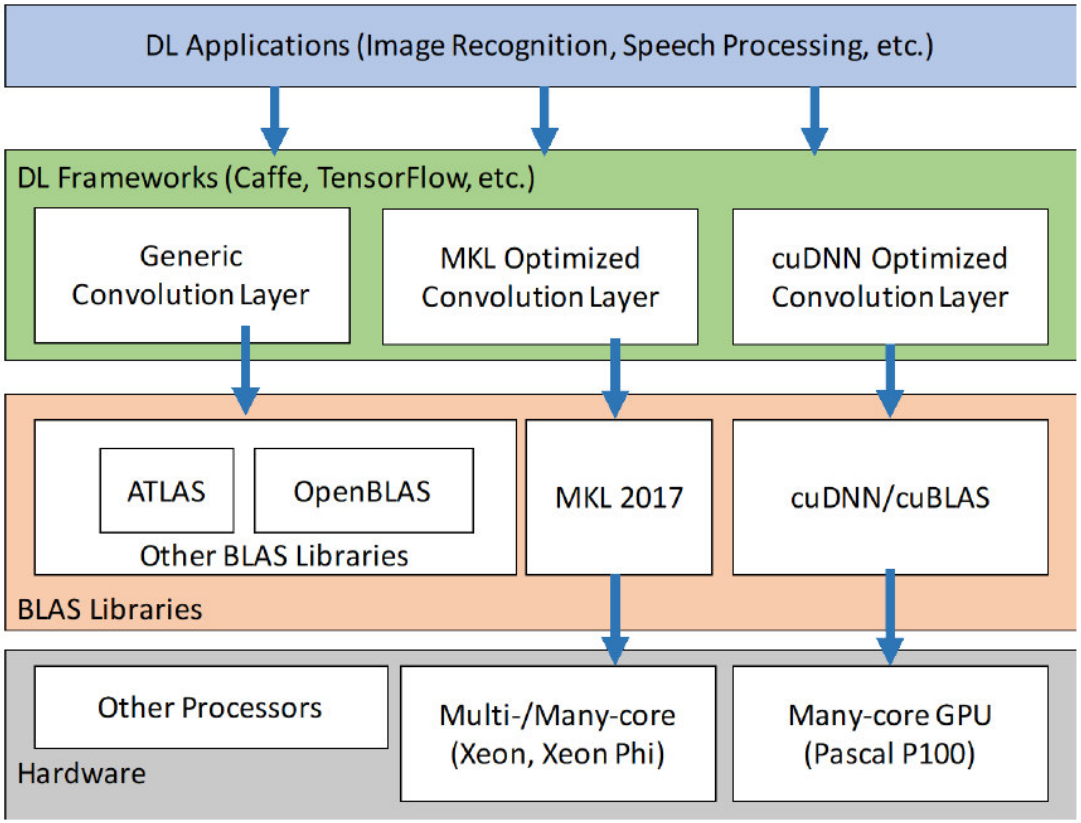


Cumulative GitHub stars by AI library, not including TensorFlow (2015—2019)
Source: Github, 2019.



https://aiindex.org

DL framework outline

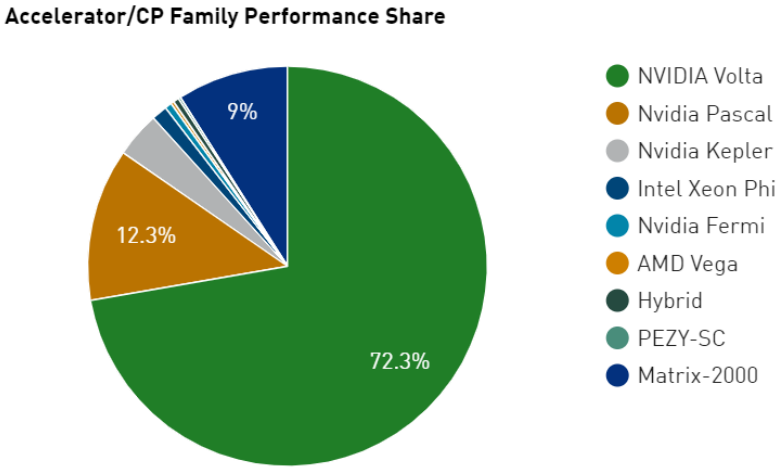


A. A. Awan, H. Subramoni, and Dhabaleswar K. Panda. "An In-depth Performance Characterization of CPU- and GPU-based DNN Training on Modern Architectures", In Proceedings of the Machine Learning on HPC Environments (MLHPC'17). ACM, New York, NY, USA, Article 8.

DL and HPC architectures

SHARCNET™
GENERAL INTEREST SEMINAR

NVIDIA GPUs are the main driving force for faster training DL models



https://top500.org

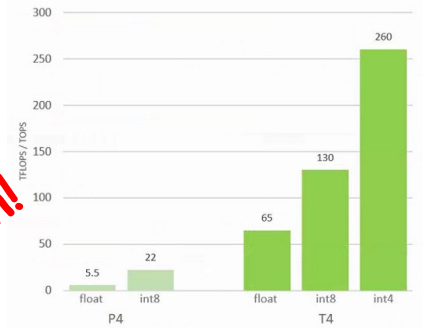
compute · calcul ONTARIO

NVIDIA T4 Turing GPU on Graham

Latest addition!

Streaming Multiprocessor
8 Tensor cores

40 SM in T4
8.1 Tflops FP32
65 Tflops FP16



compute canada | calcul canada

GPU resources in Compute Canada

As of Feb, 2020

	# of nodes	GPU type	Note
Graham	160	P100 Pascal	--gres=gpu:1
	7	V100 Volta	CPU/GPU ≤ 3.5 --gres=gpu:v100:1
	36	T4 Turing (DL target)	CPU/GPU ≤ 3.5 --gres=gpu:t4:2
Cedar	146	P100 Pascal	--gres=gpu:1
Beluga	172	V100 Volta	CPU/GPU ≤ 3.5 --gres=gpu:v100:1
Niagara	None		

Virtual environment

Allows users to create virtual environments so that one can install Python modules easily

Many versions of same module are possible

```
[isaac@gra-login3 ~]$ module load python
[isaac@gra-login3 ~]$ module list

Currently Loaded Modules:
  1) nixpkgs/16.09   (S)      3) gcccore/.5.4.0  (H)   5) ifort/.2016.4.258 (H)   7) openmpi/2.1.1 (m)   9) python/3.7.4 (t)
  2) imkl/11.3.4.258 (math)   4) icc/.2016.4.258 (H)   6) intel/2016.4      (t)   8) StdEnv/2016.4 (S)

  Where:
   S:     Module is Sticky, requires --force to unload or purge
   m:     MPI implementations / Implémentations MPI
   math:  Mathematical libraries / Bibliothèques mathématiques
   t:     Tools for development / Outils de développement
   H:               Hidden Module



[isaac@gra-login3 ~]$ virtualenv --no-download ~/tf5
Using base prefix '/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/python/3.7.4'
New python executable in /home/isaac/tf5/bin/python
Installing setuptools, pip, wheel...
done.
[isaac@gra-login3 ~]$ source tf5/bin/activate
(tf5) [isaac@gra-login3 ~]$ deactivate
[isaac@gra-login3 ~]$
```


Python packages for Tensorflow/PyTorch

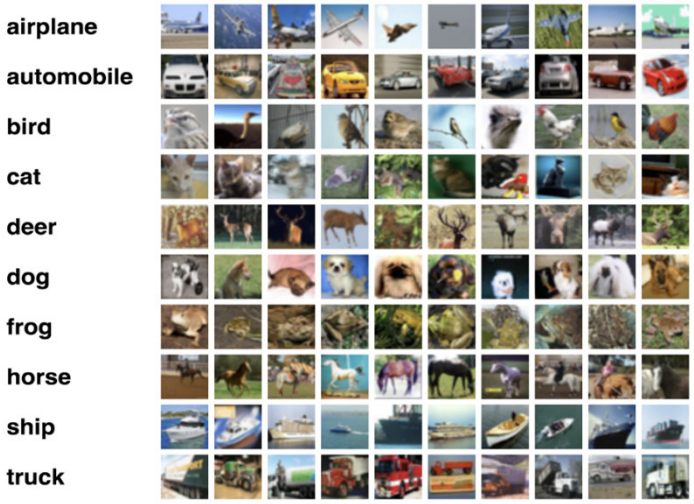
```
[isaac@cedar1 ~]$ avail_wheels "*tensor*"
name                    version    build    python    arch
----------------------  ---------  -------  --------  -------
tensorboard             2.1.0              py3       generic
tensorboardX            1.9                py2.py3   generic
tensorflow_cpu          2.1.0              cp37      generic
tensorflow_estimator    2.1.0              py2.py3   generic
tensorflow_gpu          2.1.0              cp37      generic
tensorflow_tensorboard  1.5.1              py3       generic
```

```
[isaac@cedar1 ~]$ avail_wheels "*torch*"
name                    version    build    python    arch
----------------------  ---------  -------  --------  -------
pytorch_pretrained_bert  0.6.1              py3       generic
pytorch_transformers    1.1.0              py3       generic
torch                   1.4.0              cp37      generic
torch_cluster           1.4.5              cp37      generic
torch_cpu               1.0.0              cp37      avx2
torch_geometric         1.4.2              py3       generic
torch_gpu               1.0.0              cp37      avx2
torch_scatter           2.0.3              cp37      generic
torch_sparse            0.5.1              cp37      generic
torch_spline_conv       1.1.1              cp37      generic
torchaudio              0.2        7e15d2f cp37      generic
torchnet                0.0.4              py3       generic
torchtext               0.3.1              py3       generic
torchvision             0.4.0              cp37      generic
```

Demo: Running DL interactively

CIFAR-10 dataset

The dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class.
There are 50000 training images and 10000 test images.



CIFAR10 – https://www.cs.toronto.edu/~kriz/cifar.html

SBATCH Scripts

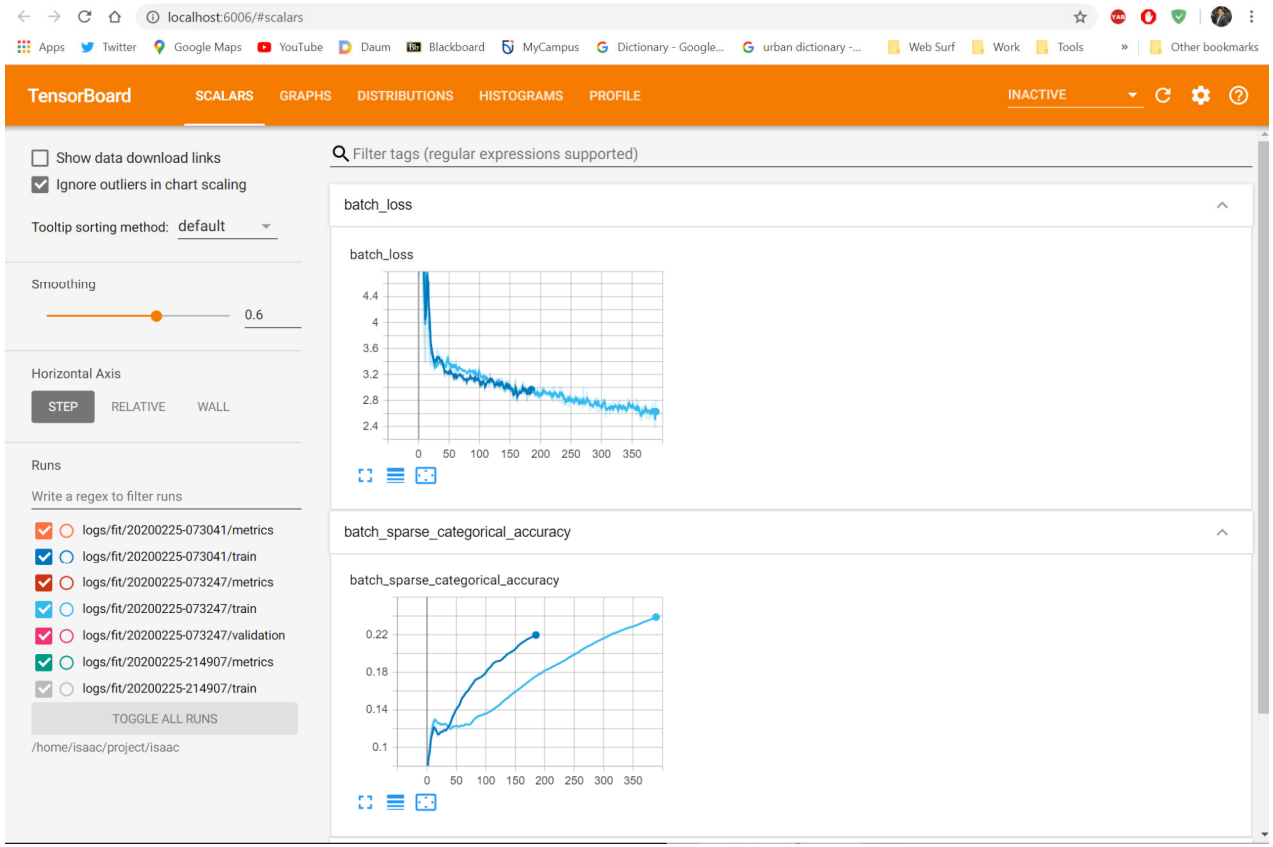
TensorFlow

```
#!/bin/bash
#SBATCH --gres=gpu:1         # request GPU "generic resource"
#SBATCH --cpus-per-task=6    # maximum CPU cores per GPU request: 6 on Cedar, 16 on Graham.
#SBATCH --mem=32000M         # memory per node
#SBATCH --time=0-03:00       # time (DD-HH:MM)
#SBATCH --output=%N-%j.out   # %N for node name, %j for jobID
#SBATCH --account=def-isaac

module load cuda cudnn
source /home/$USER/tf/bin/activate
cd ~/project/isaac
python resnet_cifar.py
```

PyTorch

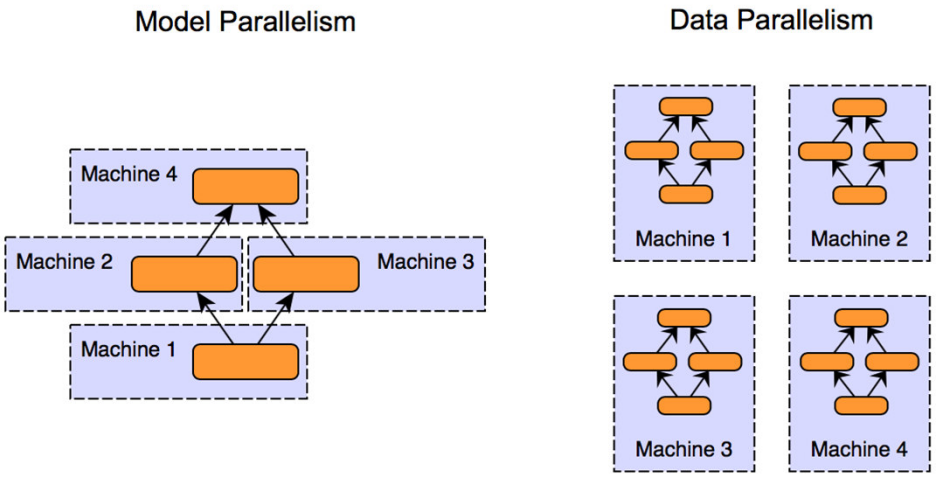
```
#!/bin/bash
#SBATCH --gres=gpu:1         # request GPU "generic resource"
#SBATCH --cpus-per-task=6    # maximum CPU cores per GPU request: 6 on Cedar, 16 on Graham.
#SBATCH --mem=32000M         # memory per node
#SBATCH --time=0-03:00       # time (DD-HH:MM)
#SBATCH --output=%N-%j.out   # %N for node name, %j for jobID
#SBATCH --account=def-isaac

source /home/$USER/tf/bin/activate
cd ~/project/isaac
python cifar10_tutorial.py
```

Tensorboard

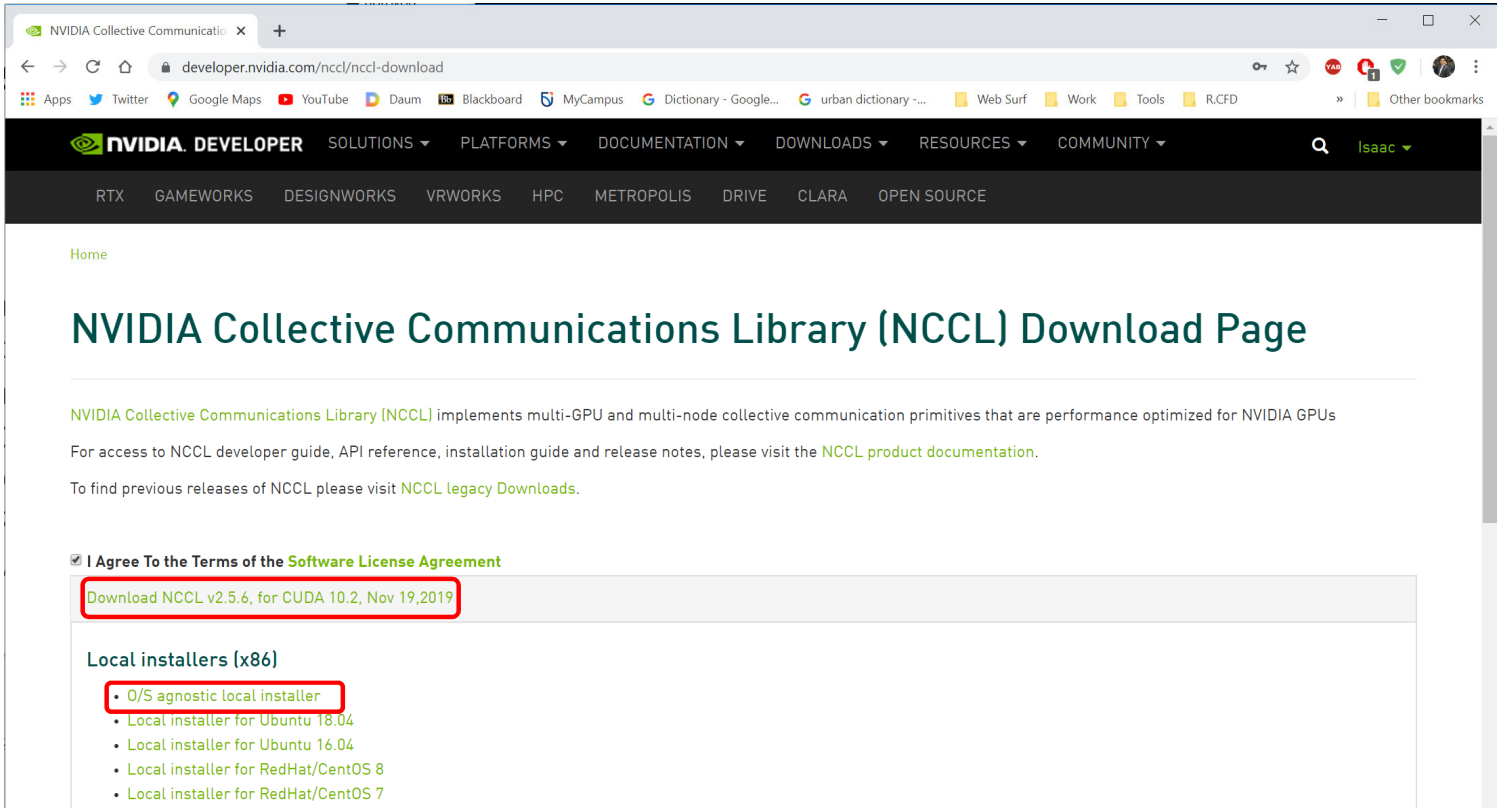


HOROVOD (Multi-GPU Multi-node)

Open source distributed DL framework for TensorFlow, PyTorch, Keras and etc.
Make distributed DL fast and easy to use
Ring-allreduce with only a few lines of modification
Developed by Uber



NCCL: NVIDIA Collective Communications Library



HOROVOD : Installation

1. Modules

```
[isaac@cedar1 ~]$ module load gcc/8.3.0 python cuda cudnn scipy-stack/2019a openmpi/4.0.1

Lmod is automatically replacing "intel/2016.4" with "gcc/8.3.0".


The following have been reloaded with a version change:
  1) gcccore/.5.4.0 => gcccore/.8.3.0     2) openmpi/2.1.1 => openmpi/4.0.1

[isaac@cedar1 ~]$ module list

Currently Loaded Modules:
  1) nixpkgs/16.09   (S)      3) StdEnv/2016.4  (S)    5) gcc/8.3.0     (t)    7) cudacore/.10.1.243 (H)    9) cudnn/7.6.5       (math)   11) ucx/1.5.2
  2) imkl/11.3.4.258 (math)   4) gcccore/.8.3.0 (H)    6) python/3.7.4  (t)    8) cuda/10.1          (t)   10) scipy-stack/2019a (math)   12) openmpi/4.0.1 (m)
```

HOROVOD : Installation

2. NCCL2

```
[isaac@cedar5 ~]$ tar xf nccl_2.5.6-2+cuda10.2_x86_64.txz
[isaac@cedar5 ~]$ ls
abc.test  nccl_2.5.6-2+cuda10.2_x86_64      nearline  projects  tensorflow           tf
log1      nccl_2.5.6-2+cuda10.2_x86_64.txz  project   scratch   TensorFlow2-tutorial
[isaac@cedar5 ~]$ ls -l nccl_2.5.6-2+cuda10.2_x86_64
total 9
drwxr-x--- 2 isaac isaac 4096 Nov 17 19:07 include
drwxr-x--- 3 isaac isaac 4096 Nov 17 19:09 lib
-rw-r----- 1 isaac isaac 1735 Nov 17 19:07 LICENSE.txt
```

```
[isaac@cedar5 ~]$ export HOROVOD_CUDA_HOME=$CUDA_HOME
[isaac@cedar5 ~]$ export HOROVOD_NCCL_HOME=/home/isaac/nccl_2.5.6-2+cuda10.2_x86_64
[isaac@cedar5 ~]$ export HOROVOD_GPU_ALLREDUCE=NCCL
[isaac@cedar5 ~]$ env |grep HOROVOD
HOROVOD_CUDA_HOME=/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243
HOROVOD_NCCL_HOME=/home/isaac/nccl_2.5.6-2+cuda10.2_x86_64
HOROVOD_GPU_ALLREDUCE=NCCL
```

HOROVOD : Installation

3. Install horovod

```
[isaac@cedar1 isaac]$ source ~/tf/bin/activate
(tf) [isaac@cedar1 isaac]$ pip install --no-cache-dir horovod
Ignoring pip: markers 'python_version < "3"' don't match your environment
Looking in links: /cvmfs/soft.computecanada.ca/custom/python/wheelhouse/avx2, /cvmfs/soft.computecanada.ca/custom/python/wheelhouse/generic
Collecting horovod
  Downloading https://files.pythonhosted.org/packages/e8/1b/51c36afc6dab9afad1caed2a11366d1e238e42f3c51177d06779356f8f1a/horovod-0.19.0.tar.gz (2.9MB)
     |████████████████████████████████| 2.9MB 9.7MB/s
Requirement already satisfied: cloudpickle in /home/isaac/tf/lib/python3.7/site-packages (from horovod) (1.3.0)
Requirement already satisfied: psutil in /home/isaac/tf/lib/python3.7/site-packages (from horovod) (5.6.5)
Requirement already satisfied: pyyaml in /home/isaac/tf/lib/python3.7/site-packages (from horovod) (5.1.2)
Requirement already satisfied: six in /home/isaac/tf/lib/python3.7/site-packages (from horovod) (1.14.0)
Requirement already satisfied: cffi>=1.4.0 in /home/isaac/tf/lib/python3.7/site-packages (from horovod) (1.13.2)
Requirement already satisfied: pycparser in /home/isaac/tf/lib/python3.7/site-packages (from cffi>=1.4.0->horovod) (2.19)
Building wheels for collected packages: horovod
  Building wheel for horovod (setup.py) ... done
  Stored in directory: /tmp/pip-ephem-wheel-cache-cpmgcf_d/wheels/2a/09/5f/bf4132f534447b5b4b110921ef88d93cb14df3e8e9c1905c2c
Successfully built horovod
Installing collected packages: horovod
Successfully installed horovod-0.19.0
(tf) [isaac@cedar1 isaac]$
```

HOROVOD : SBATCH

```
#!/bin/bash
#SBATCH --gres=gpu:1         # request GPU "generic resource"
#SBATCH --cpus-per-task=12   # maximum CPU cores per GPU request: 6 on Cedar, 16 on Graham.
#SBATCH --mem=12000M         # memory per node
#SBATCH --time=0-01:00       # time (DD-HH:MM)
#SBATCH --output=%N-%j.out   # %N for node name, %j for jobID
#SBATCH --account=def-isaac

module load gcc/8.3.0 python cuda cudnn scipy-stack/2019a openmpi/4.0.1

source /home/$USER/tf/bin/activate
cd /home/isaac/project/isaac/benchmarks

NCCL_DEBUG=INFO
srun python /home/isaac/project/isaac/benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py\
        --model resnet50\
        --batch_size 32\
        --variable_update horovod\
```


compute
calcul
ONTARIO


compute
canada
calcul
canada

Thanks!


SHARCNET™