Hewlett Packard
Enterprise

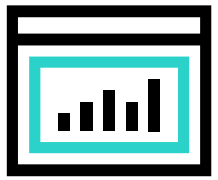
Navigating through the A.I. Landscape

Will Vick
AI Business Development
Technical Strategy
HPC/AI Competency Center

AI ecosystem

A working solution requires every layer of this stack

Deep learning ecosystem



Software

Caffe

Keras

theano

torch

KALDI

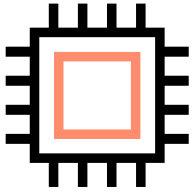
Caffe2

Microsoft CNTK

mxnet

TensorFlow

Chainer



Hardware

NVIDIA

intel

Movidius

nervana SYSTEMS

ARM

Google

XILINX

AMD

QUALCOMM

What is the optimal hardware and software configuration for my deep learning workload?

How many GPUs per node?

How many nodes in a cluster?

InfiniBand or Ethernet?

Which CPUs and how many?

Does storage matter?

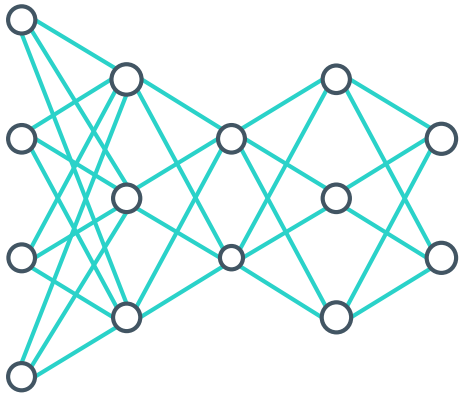
One size does NOT fit all

Application

Data type

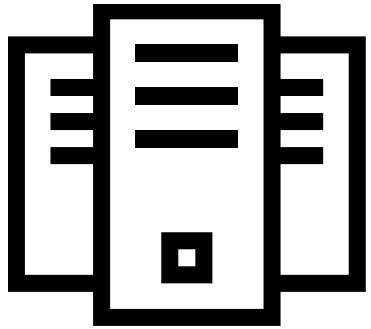
Data size

Model
(topology of artificial neural network)



- How many layers
- How many neurons per layer
- Connections between neurons (types of layers)

Optimal hardware and software



Deep Learning Frameworks

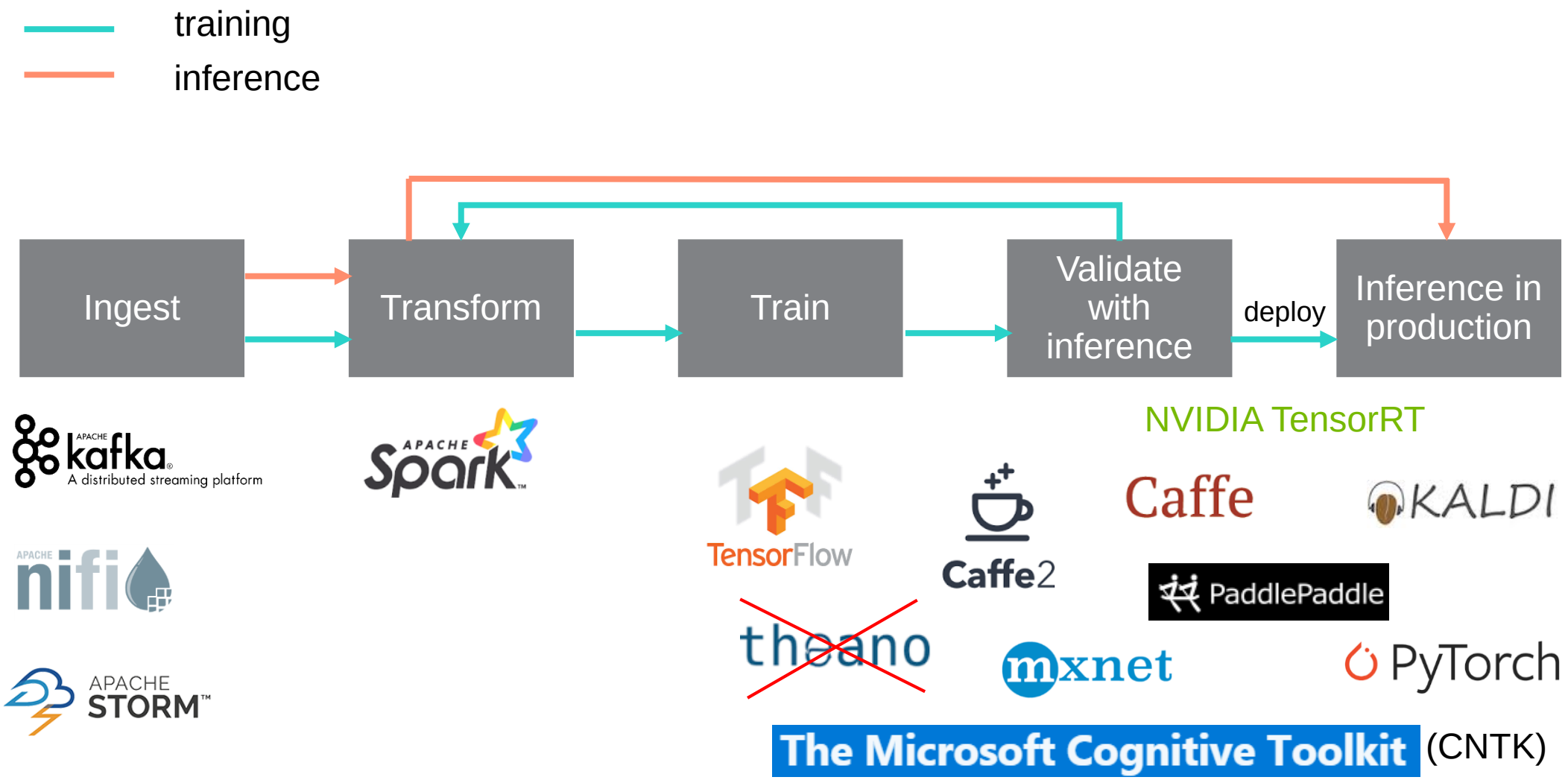
– Frameworks are really the '**schedulers**' for the Deep Learning workload
  – It connect to (almost) any kind of data : Video, Images, Sounds, Signals, Text, time series,…
  – in (virtually) any format : available on a filesystem, in DB format, Big data (Kafka, Spark,…)

– They are **programmed with scripts** with high level languages
  – Python for TensorFlow,
  – Lua
  – C++

– Or can **interact** with Data Scientists Tools
  – Matlab
  – R

```
import os
import tensorflow as tf
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.optimizers import SGD
from keras.utils import multi_gpu_model

# Generate dummy data
x_train = np.random.random((examples, channels, dx, dy))
y_train = keras.utils.to_categorical(np.random.randint(10, size=(examples, 1)), num_classes=10)
x_test = np.random.random((examples, channels, dx, dy))
y_test = keras.utils.to_categorical(np.random.randint(10, size=(examples, 1)), num_classes=10)

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)

model = Sequential()
model.add(Conv2D(planes, (1, 1), activation='relu', input_shape=(channels, dx, dy))
for i in range (layers):
    model.add(Conv2D(planes, (3, 3), activation='relu', padding = 'same', )
model.add(Conv2D(1, (1, 1), activation='relu', padding = 'same', )
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(10, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer=sgd)

model.fit(x_train, y_train, batch_size=minibatch, epochs=1)
```

Phases of machine learning



Most popular frameworks

Software	Affiliated company	Supported HW	Written in	Interface	Good for
TensorFlow	Google	x86 NVIDIA GPUs AMD GPUs	C++, Python	Python, C++, Java, Go, Swift	All use cases
PyTorch		x86 NVIDIA GPUs	C++	Python	Natural language processing
mxnet	APACHE SOFTWARE FOUNDATION amazon	x86 NVIDIA GPUs	C++, Python	Python, C++, Scala, Julia, Perl, R	All use cases
Caffe	BVLC	x86 NVIDIA GPUs AMD GPUs	C++	Python, bash	Image processing
KALDI		x86 NVIDIA GPUs	C++	bash	Speech recognition

https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software

Deep learning frameworks and their dependencies

UI, development tools	 NVIDIA DIGITS (Caffe, Torch, TensorFlow)
High-level APIs	Keras (TensorFlow, CNTK, Theano, MXNet) **TF Layers** (TensorFlow) **Brew** (Caffe2) GLUON (MXNet)
Deep learning and machine learning frameworks	TensorFlow Caffe CNTK PyTorch theano Caffe2 BigDL mxnet KALDI scikit learn data iku H_2O.ai
Hardware-specific libraries for basic operations for deep neural networks (BLAS + FFT, convolutions, etc)	cuDNN Intel® MKL-DNN **MIOpen**
Optimized linear algebra libraries, many support BLAS interface, hardware specific	cuBLAS, MKL, OpenBLAS, rocBLAS, MIOpenGEMM
Accelerator-specific drivers and software	NVIDIA drivers, CUDA, ROCm

The Zoology of Neural networks

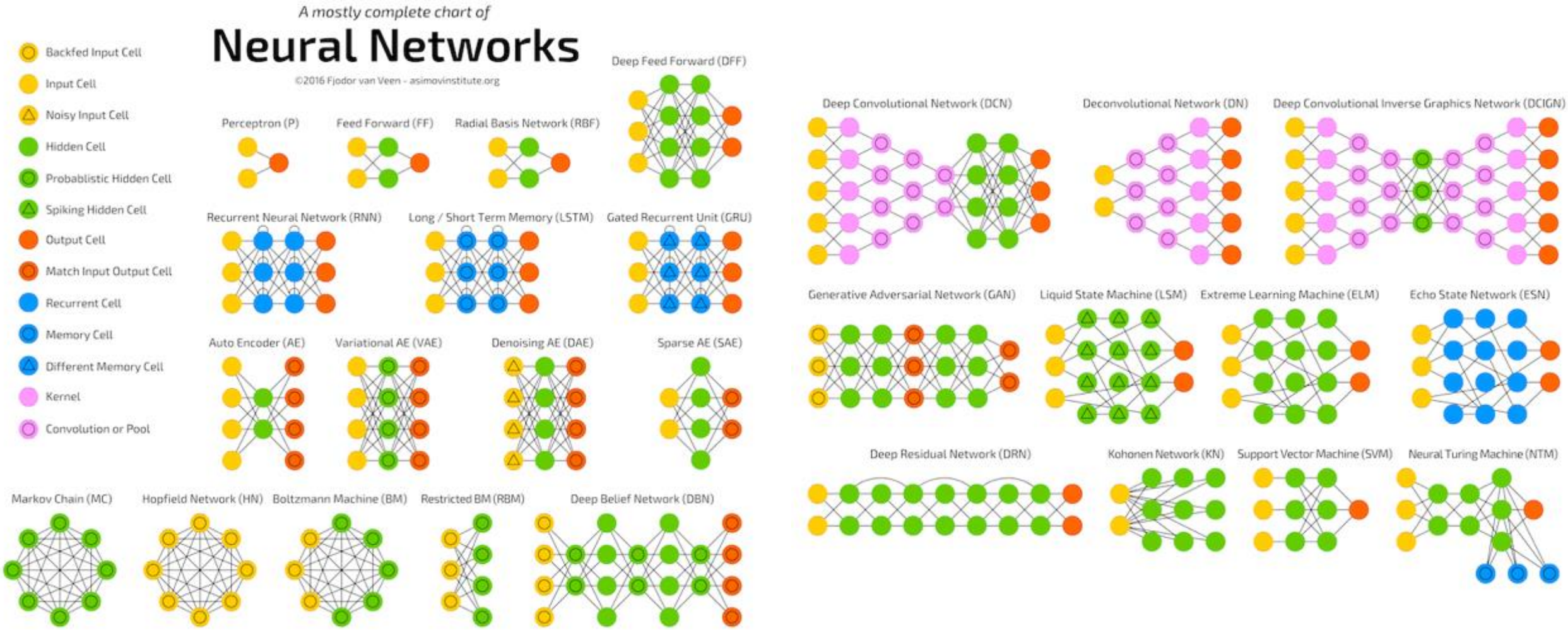
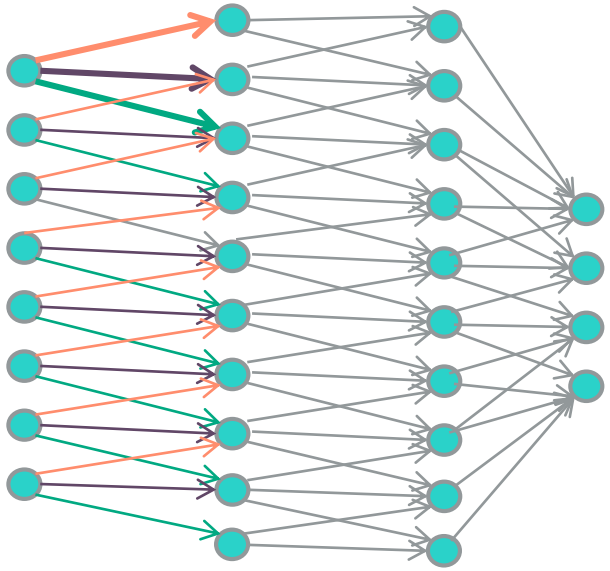


Image source: http://www.asimovinstitute.org/neural-network-zoo/

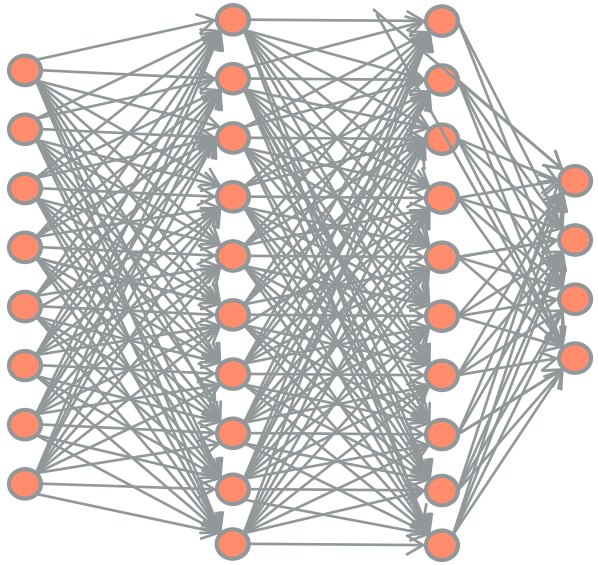
Types of artificial neural networks

Topology to fit data characteristics

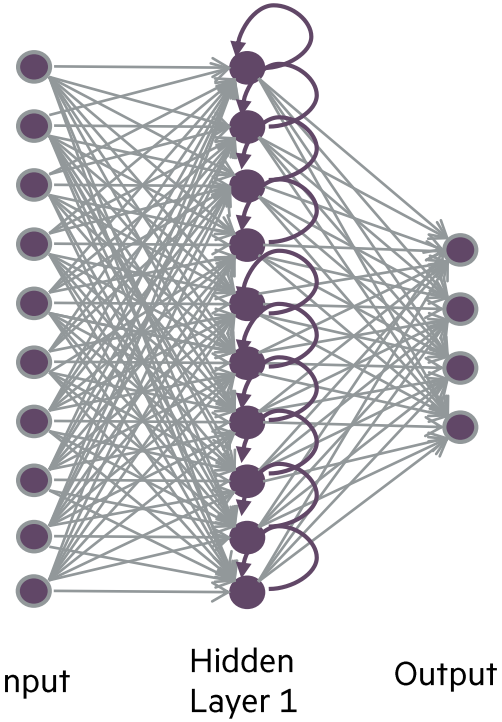
Convolutional:
Images



Fully connected:
Speech, text, sensor



Recurrent:
Speech, text, sensor



Convolutional Neural networks

Scope : Mainly used for images and video, could easily cover text and audio.

Typical use: Feed in images and the network that classifies the data. They can identify object, faces, cars, animals an so on.

The biological inspiration for CNNs is the visual cortex in different animals. The cells in the visual cortex are sensitive to small regions of the input. We call this the *visual field.* These smaller regions are unified together to cover all the visual field.

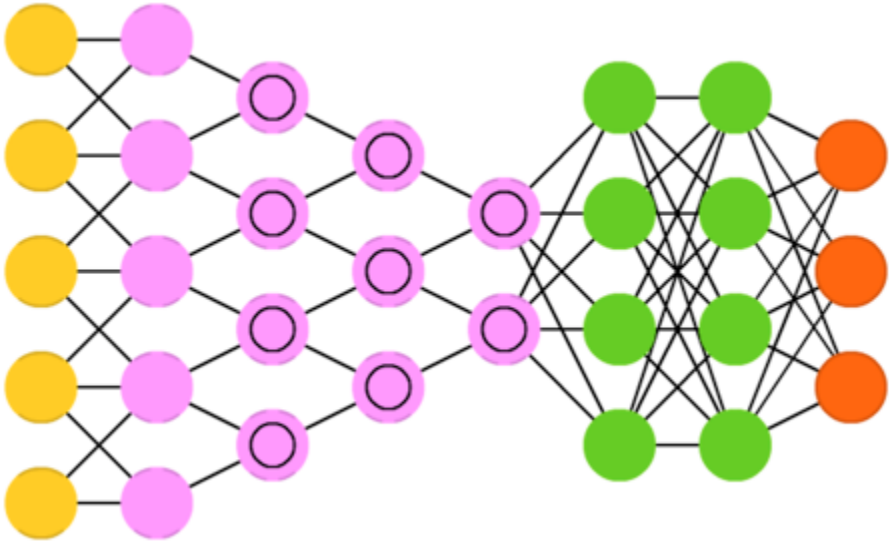
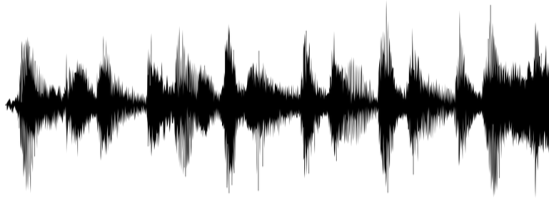
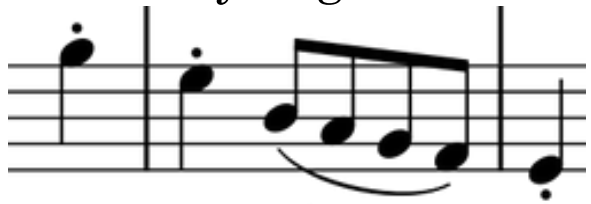


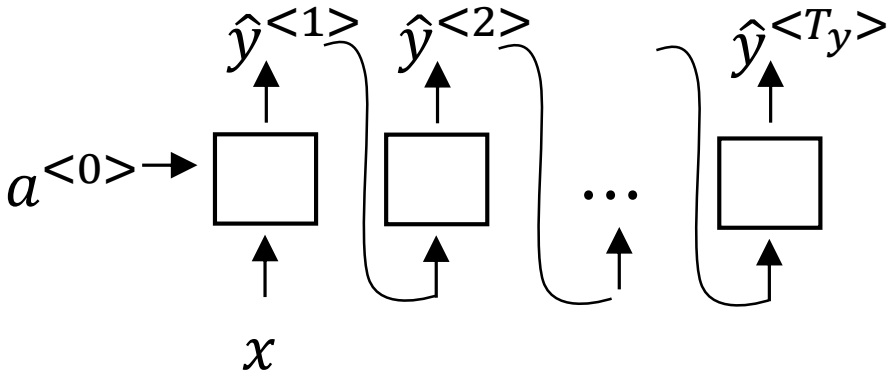
Image source: http://www.asimovinstitute.org/neural-network-zoo/

RNN-Sequence data

Speech recognition		→	"The quick brown fox jumped over the lazy dog."
Music generation	∅	→	
Sentiment classification	"There is nothing to like in this movie."	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

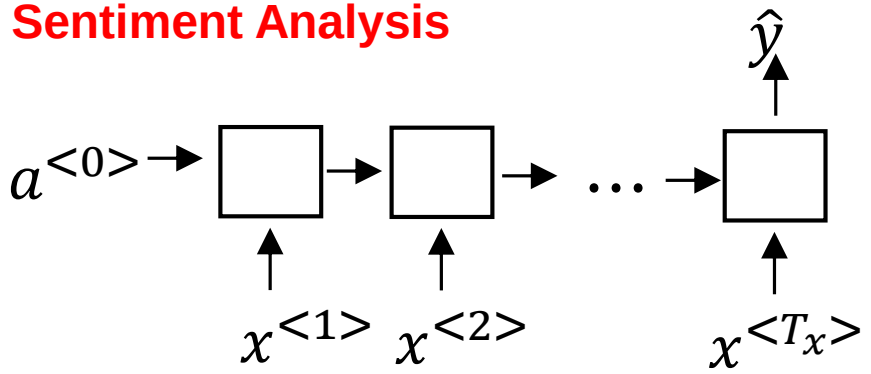
RNN types

Music Generation



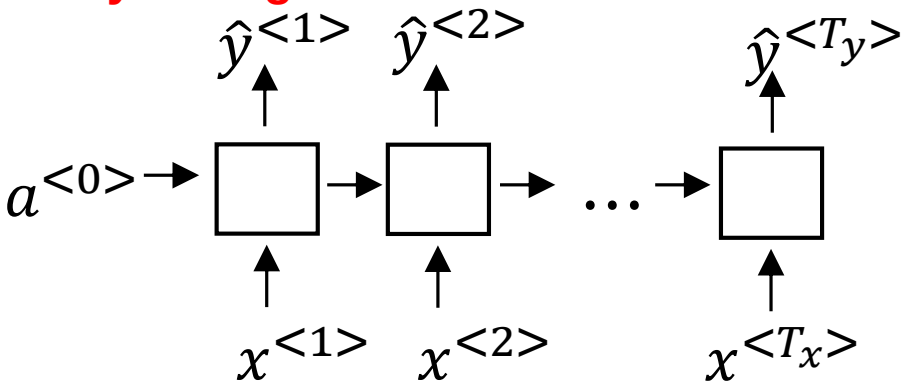
One to many

Sentiment Analysis

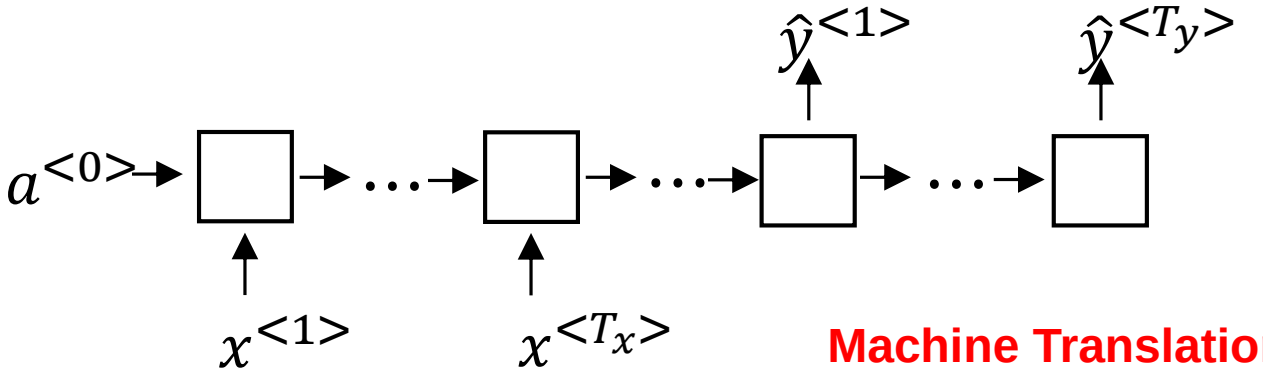


Many to one

Name Entity Recognition



Many to many



Machine Translation

Many to many

Distributed training

NCCL and NCCL 2, NVIDIA Collective Communications Library
(TensorFlow, PyTorch, MXNet, Caffe2, Microsoft Cognitive Toolkit)
https://developer.nvidia.com/nccl

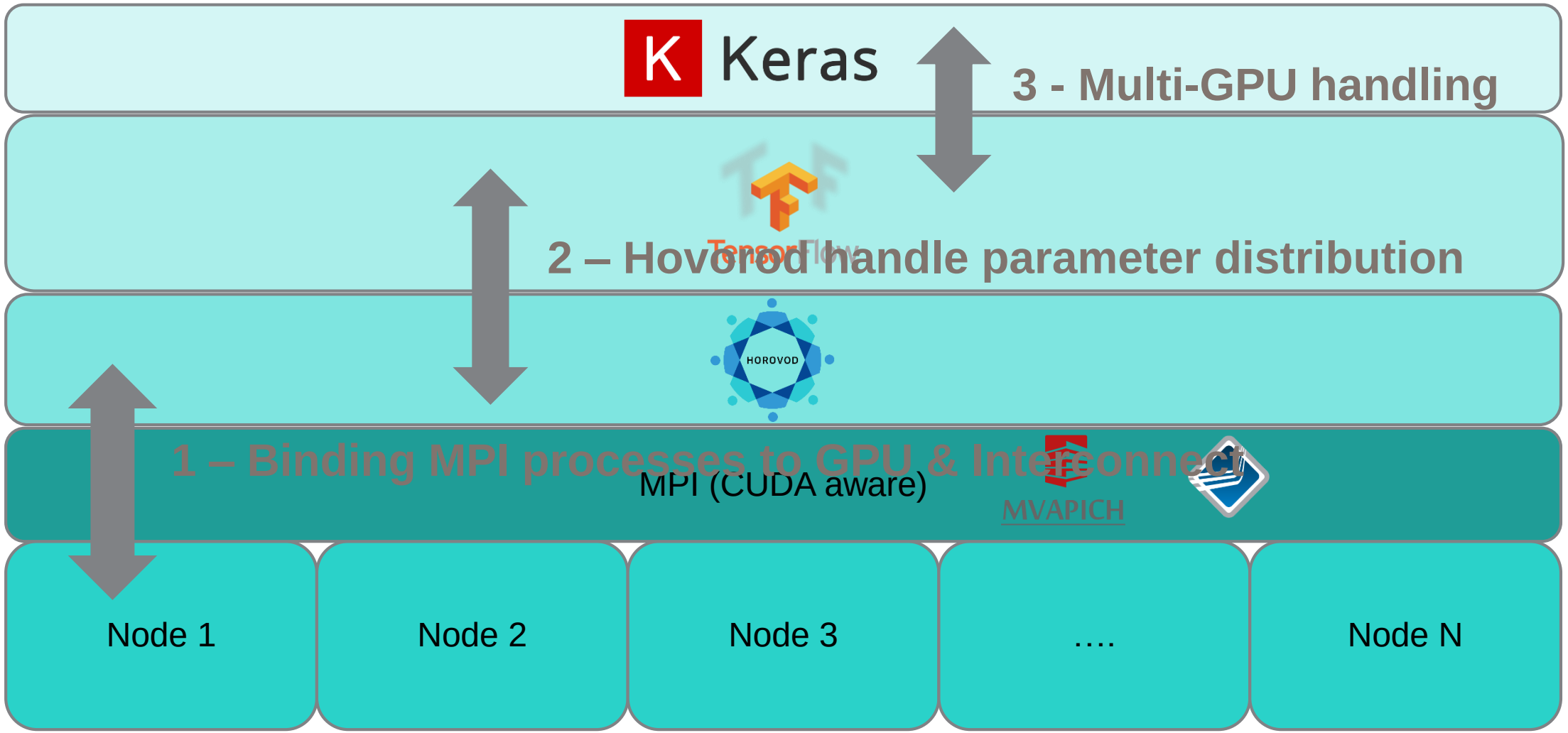
MPI-based distributed training
(TensorFlow, Keras, PyTorch)
https://github.com/uber/horovod

RPC framework
(TensorFlow)
https://grpc.io/

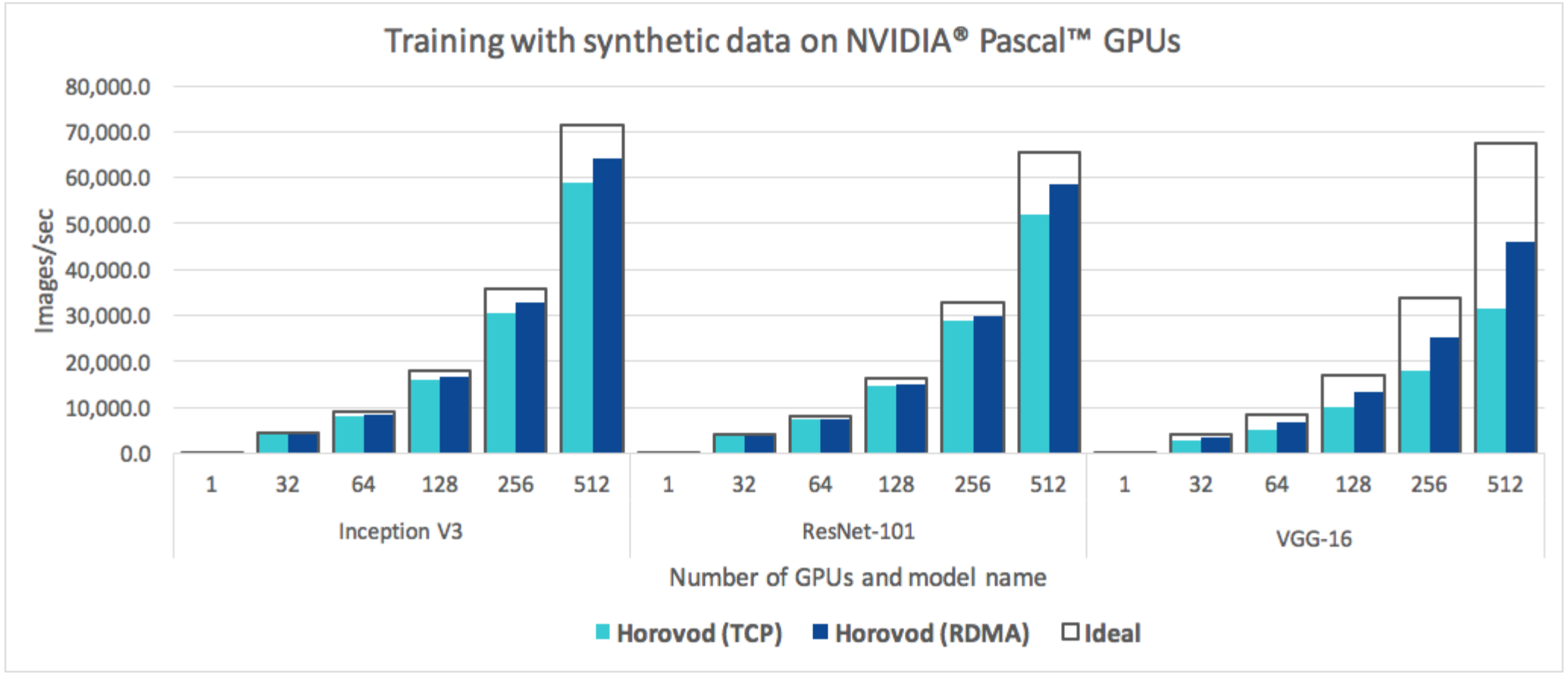
Gloo, a collective communications library
(PyTorch, Caffe2)
https://github.com/facebookincubator/gloo

MPI
OpenMPI: https://www.open-mpi.org/
MPICH: https://www.mpich.org/

Distributed training



Optimizing scaling with Horovod

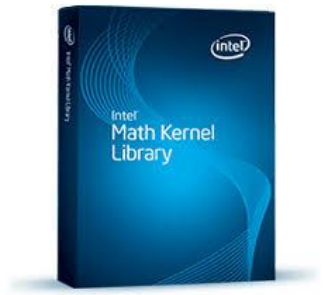


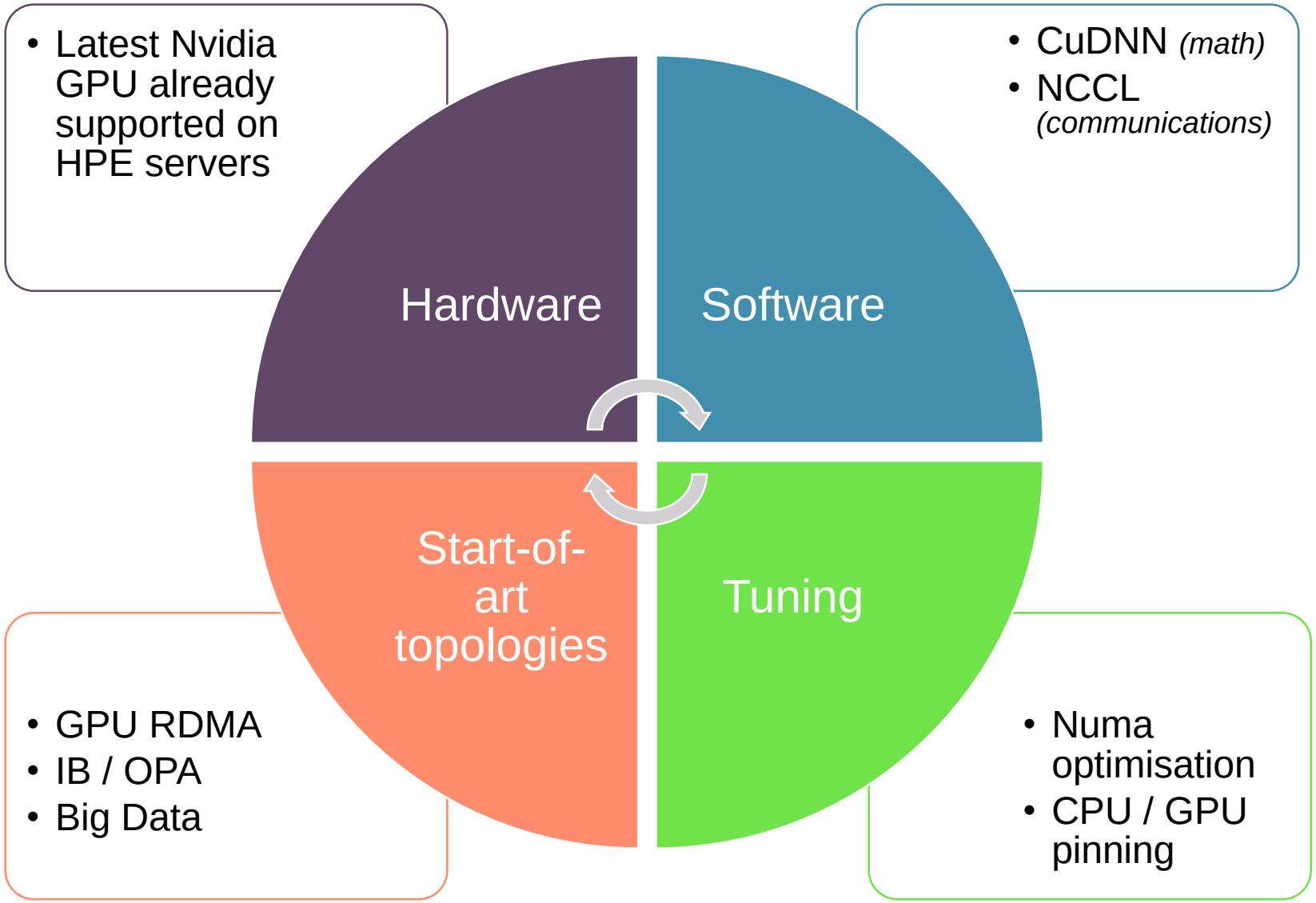
Optimized stack

– Run on professional OS :
  – RHEL, Centos, SUSE, Scientific Linux
  – Secured & patch kernel
  – Large support for enterprise architecture
    – Filesystems : Lustre ,Weka.io, GPFS, HDFS, …

– Get the most of the hardware with :
  – Optimized libraries
    – GPU : Cuda runtime
    – Math : MKL, CuBlas, CuDnn, Sci-py

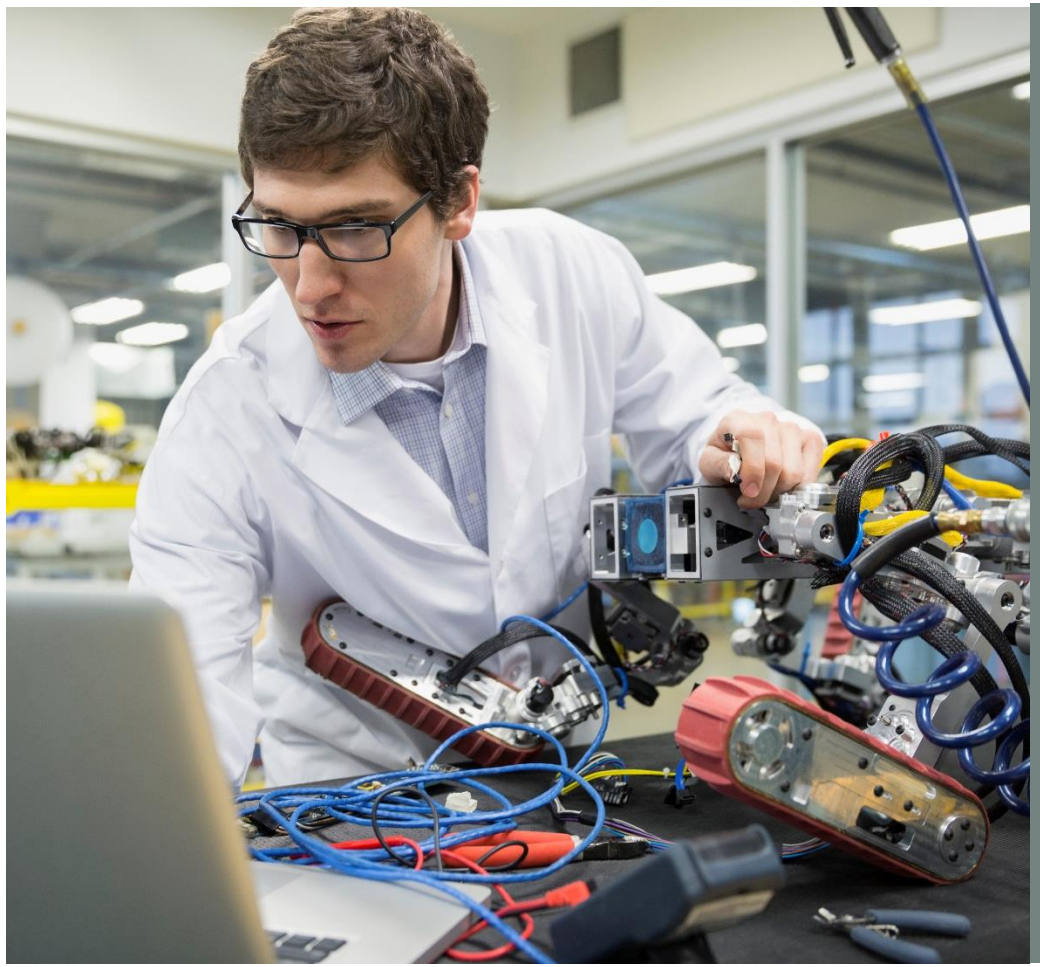
– Build Framework with state-of-art :
  – Optimized instructions set
  – Optimized GPU support




• Latest Nvidia GPU already supported on HPE servers
Hardware
Software
• CuDNN *(math)*
• NCCL *(communications)*
Start-of-art topologies
Tuning
• GPU RDMA
• IB / OPA
• Big Data
• Numa optimisation
• CPU / GPU pinning

HPE Deep Learning Cookbook

Tools to guide choice of the best hardware and software for deep learning



Eliminate the "guesswork"
when choosing hardware and software for deep learning

Validate
new hardware/software configuration

Get started fast
with technology recipes

Hewlett Packard Enterprise

https://www.hpe.com/software/dl-cookbook

HPE Deep Learning Cookbook

Main components

HPE Deep Learning Benchmarking Suite

Automated benchmarking tool to collect performance of different deep learning workloads on various hardware and software configurations.

Available on GitHub

HPE Deep Learning Performance Guide

A web-based tool to guide a choice of optimal hardware and software configuration via analysis of collected performance data and applying performance models.

http://dlpg.labs.hpe.com/

Reference Designs

Reference hardware/software stacks for particular classes of deep learning workloads.

Image Classification Reference Designs released

HPE Deep Learning Benchmarking Suite

An open source tool to benchmark DL frameworks and models

- **Frameworks**: TensorFlow, Caffe, Caffe2, MXNet, PyTorch, TensorRT
  - Frameworks can have multiple benchmark backends
- **Neural nets**: 20 models (AlexNet, GoogleNet, ResNets, VGGs …)
- **Runtimes**: bare metal / docker / singularity
- **Containers**: reference docker files / NVIDIA GPU Cloud
- **Resource monitor**: GPU/CPU/memory utilization
- **API**: command line interface / Python API

Target metrics

- Iteration time / throughput (instances / second)
- End-to-end (time to convergence)

More information

- HPE Developer Portal: https://www.hpe.com/software/dl-cookbook

Thank you!