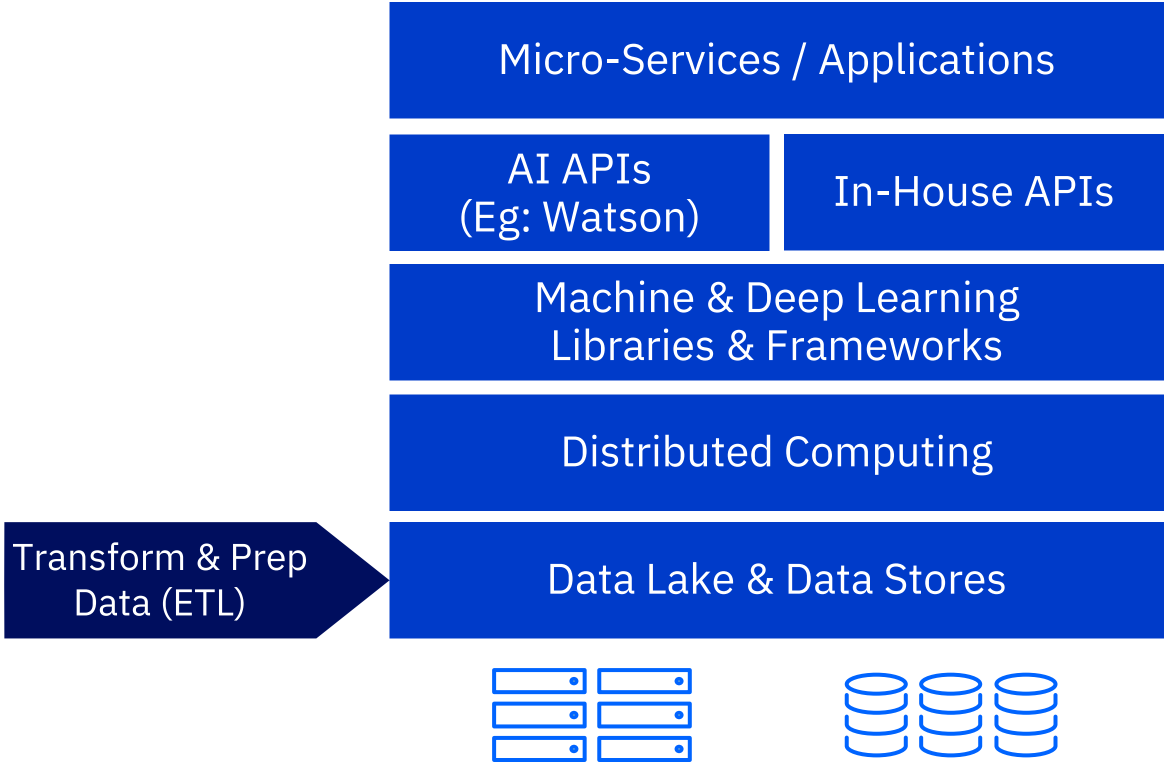
IBM PowerAI Deep Learning Platform

(architecture, hardware, software innovation)

Ing. Florin Manaila
Senior Architect and Inventor
Cognitive Systems (HPC and Deep Learning)
IBM Systems Hardware Europe
Member of the IBM Academy of Technology (AoT)

Lund, March 21, 2019

AI Infrastructure Stack



Segment Specific:
Finance, Retail, Healthcare, Automotive

Speech, Vision,
NLP, Sentiment

TensorFlow, Caffe,
Pytoch
SparkML, Snap.ML

Spark, MPI

Hadoop HDFS,
NoSQL DBs,
Parallel File System

Accelerated Infrastructure

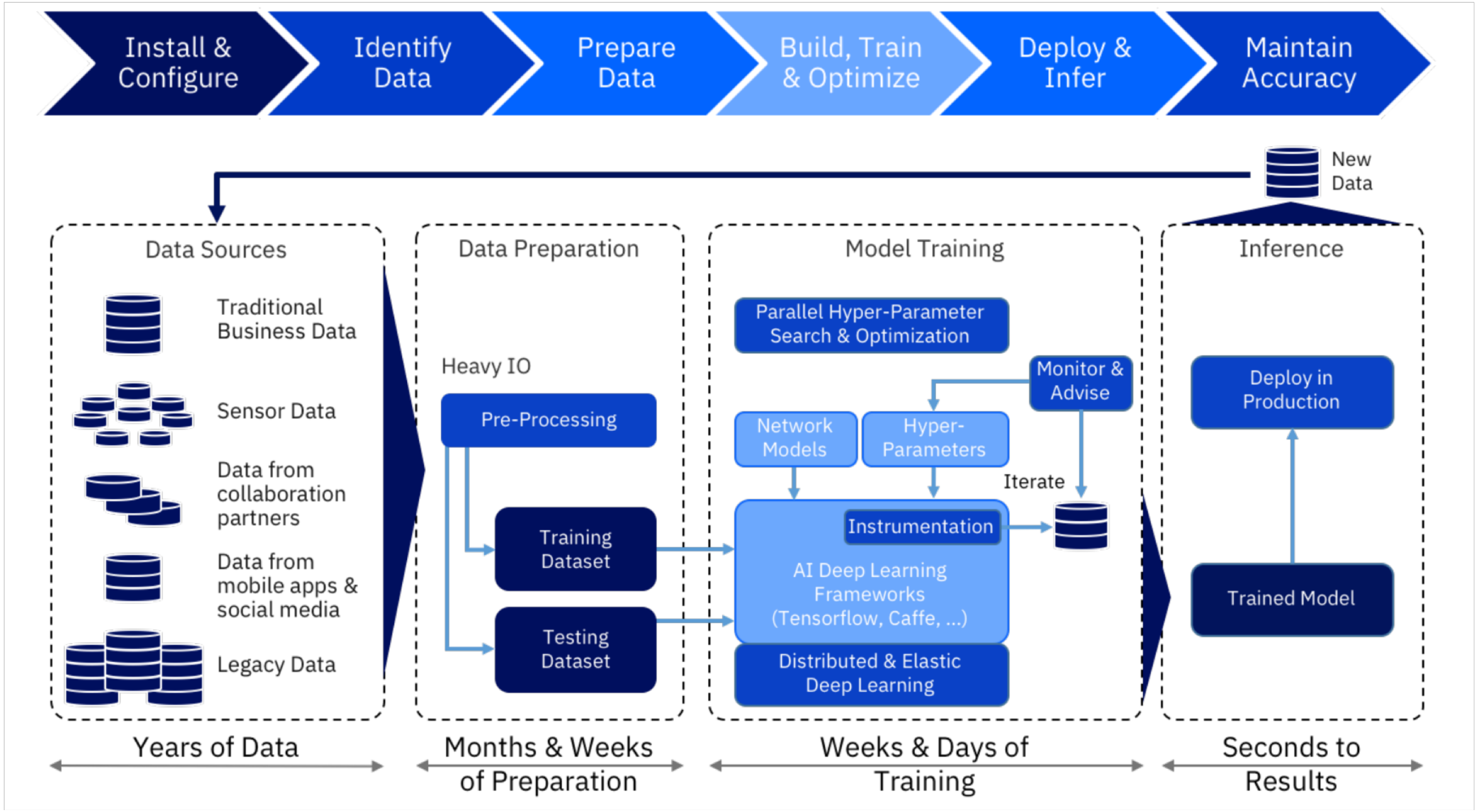
AI Infrastructure Stack Challenges

Micro-Services / Applications	Use Case Identification, Access to Enough Data
AI APIs (Eg: Watson) / In-House APIs	Finding Right “Tagged” Data, Model Integrity
Machine & Deep Learning Libraries & Frameworks	Feature extraction, Selecting Right Model, Hyper-parameter tuning
Distributed Computing	Multi-tenant, GPU Virtualization, DL Framework Scaling
Data Lake & Data Stores	Data Prep, ETL, Curation, Data Labeling

Transform & Prep Data (ETL)

Performance to Reduce Training Time

AI Workflow



Install & Configure
Identify Data
Prepare Data
Build, Train & Optimize
Deploy & Infer
Maintain Accuracy
New Data
Data Sources
Traditional Business Data
Sensor Data
Data from collaboration partners
Data from mobile apps & social media
Legacy Data
Data Preparation
Heavy IO
Pre-Processing
Training Dataset
Testing Dataset
Model Training
Parallel Hyper-Parameter Search & Optimization
Monitor & Advise
Network Models
Hyper-Parameters
Iterate
Instrumentation
AI Deep Learning Frameworks (Tensorflow, Caffe, ...)
Distributed & Elastic Deep Learning
Inference
Deploy in Production
Trained Model
Years of Data
Months & Weeks of Preparation
Weeks & Days of Training
Seconds to Results

What’s in the training of deep neural networks?

Data

Millions of images, sentences

Terabytes

Neural network model

Billions of parameters

Gigabytes

Computation

Iterative gradient based search

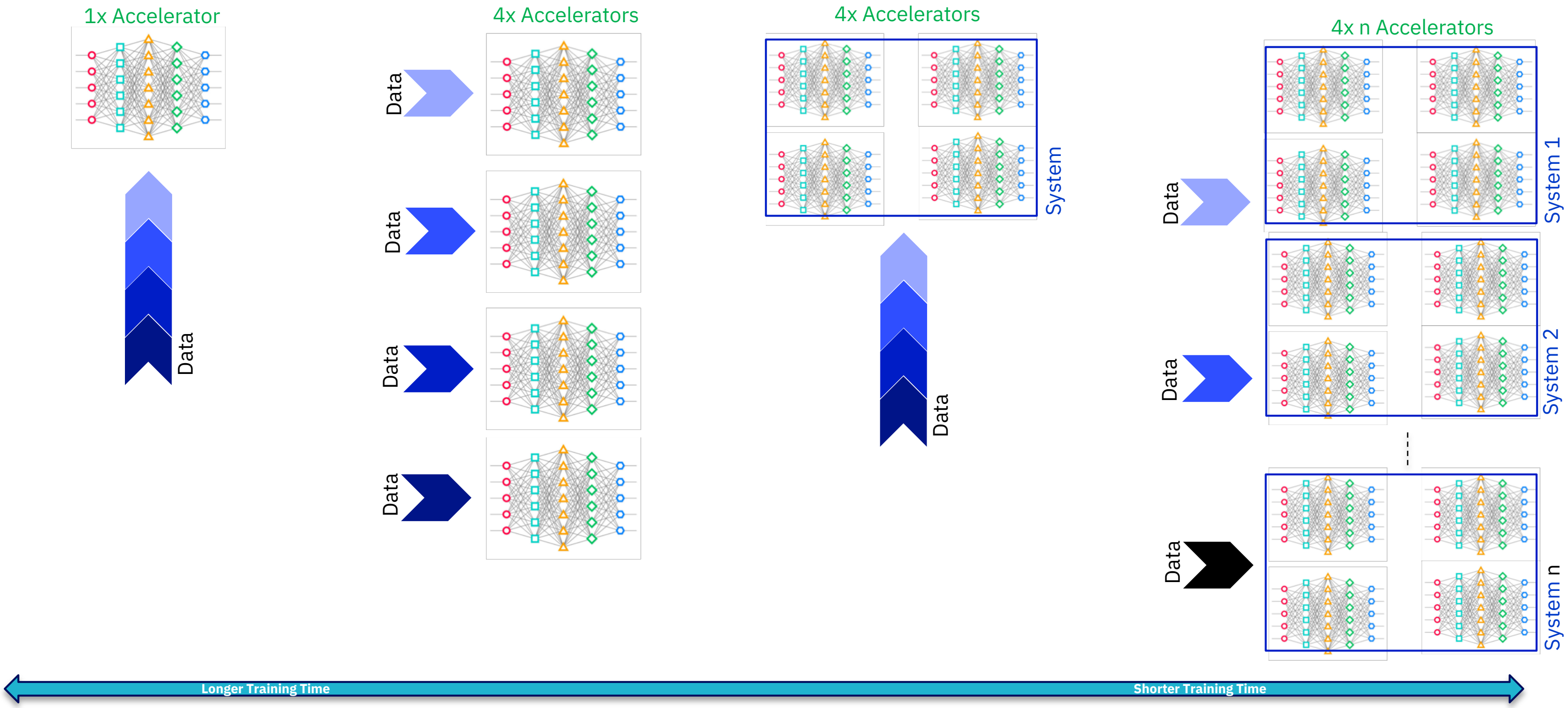
Millions of iterations
Mainly matrix operations

Workload characteristics: Both compute and data intensive!

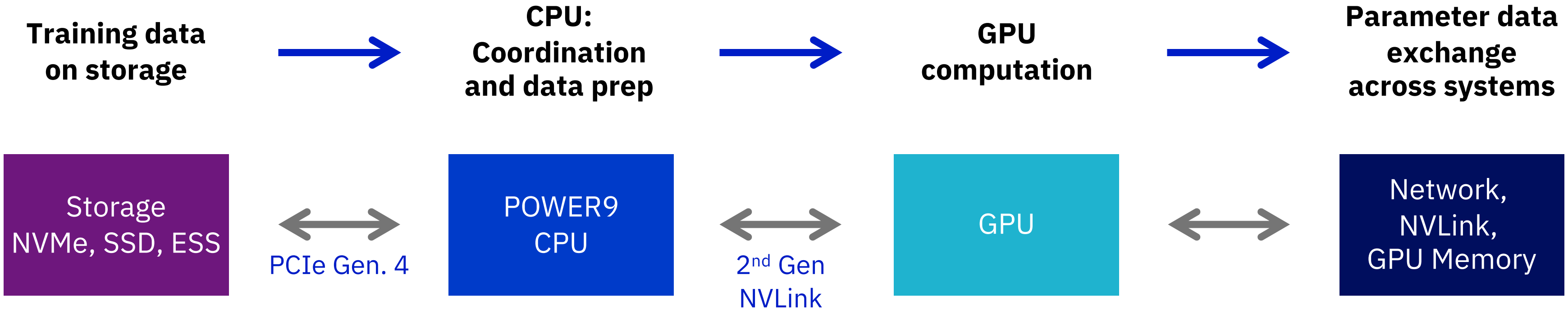
Deep Learning at work

Available options

SINGLE ACCELERATOR | DATA PARALLEL | MODEL PARALLEL | DATA AND MODEL PARALLEL



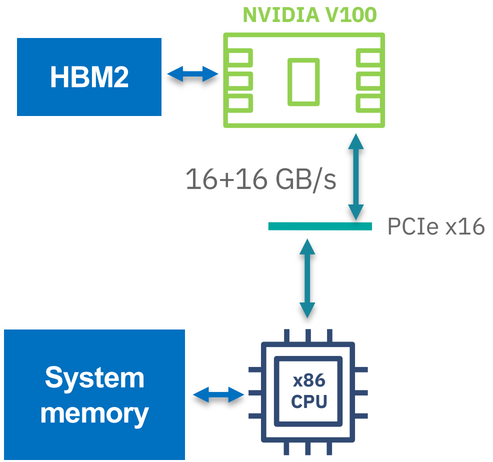
Data processing stages for distributed deep learning

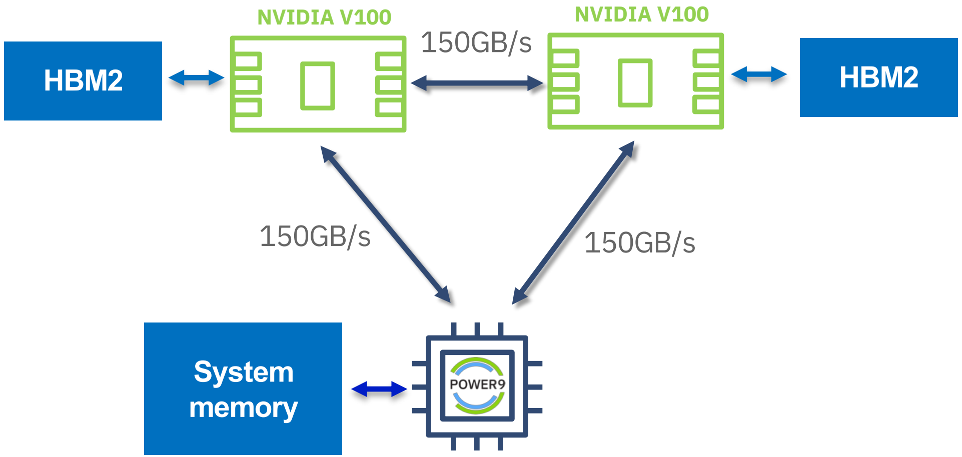


Source: Hillery Hunter, IBM, GTC 2018

NVIDIA GPU implementation in AC922 Deep Learning System

NVLINK 2.0





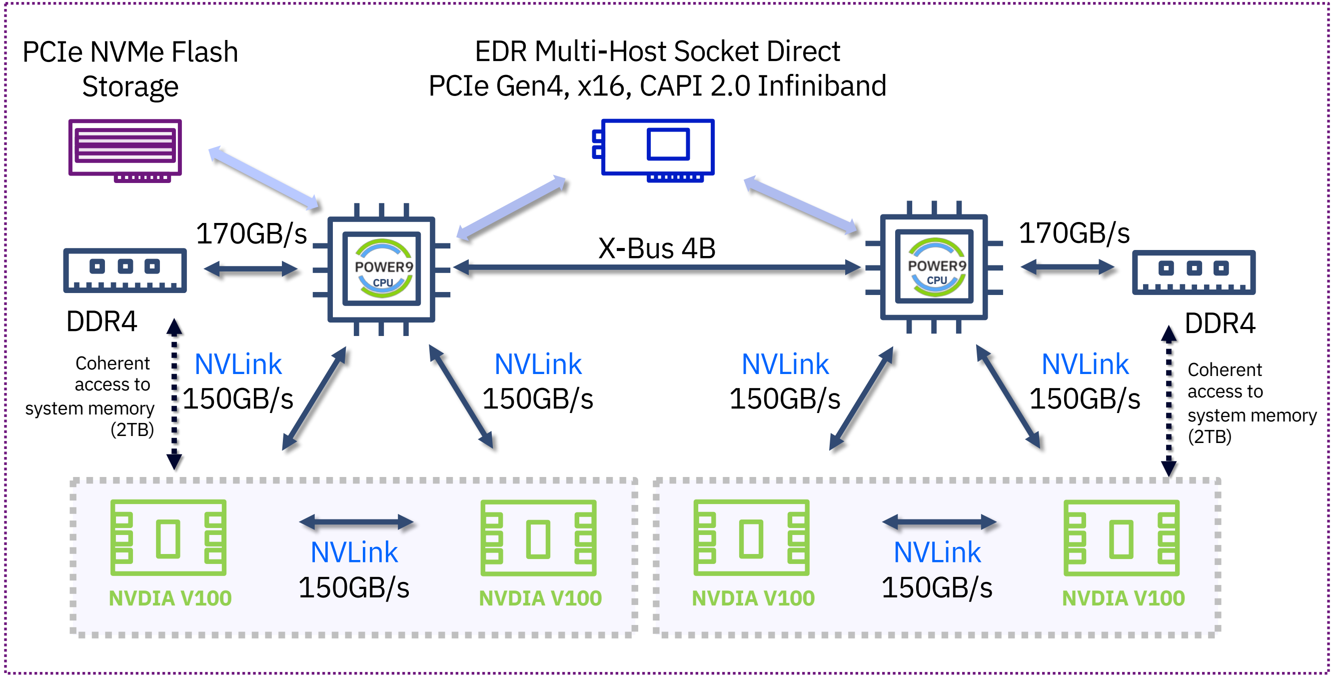
- Acceleration limited by PCIe Gen3

Innovative Systems with NVLink 2.0:

- Faster GPU-GPU communication
- Breaks down barriers between CPU and GPU
- New system architectures

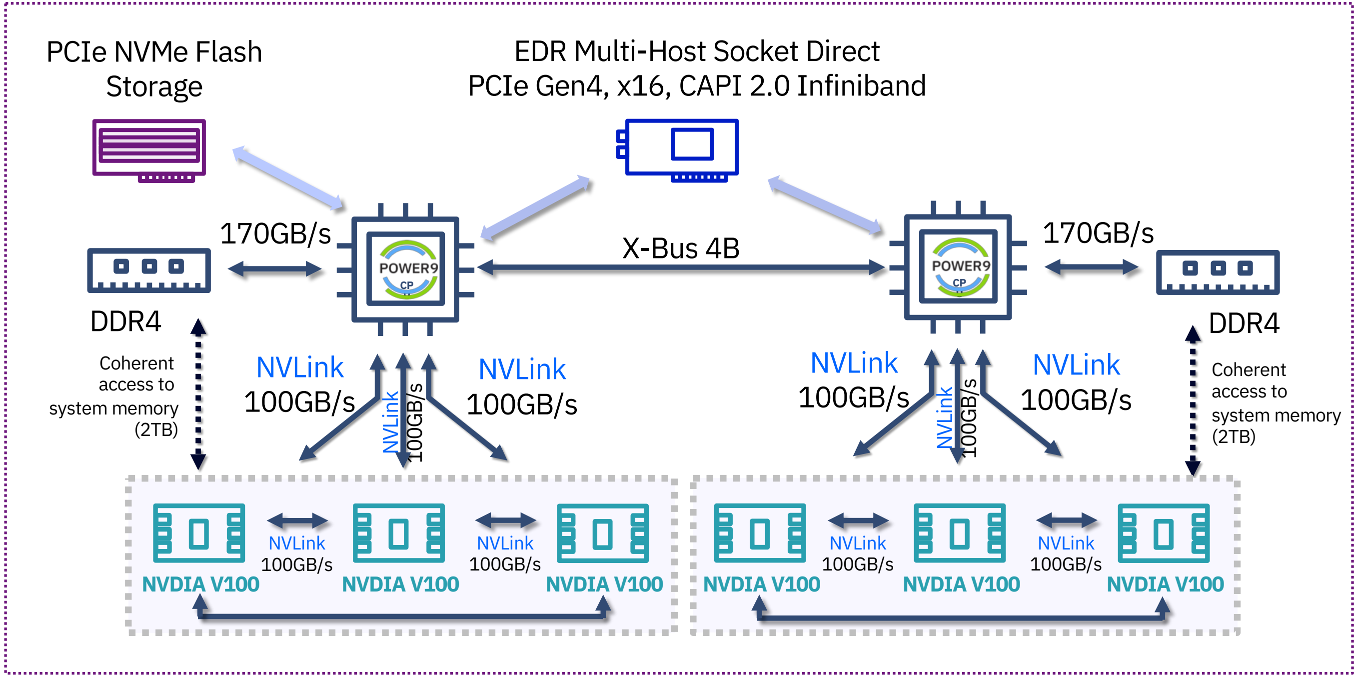
IBM AC922 Deep Learning System Architecture

AC922-GTG



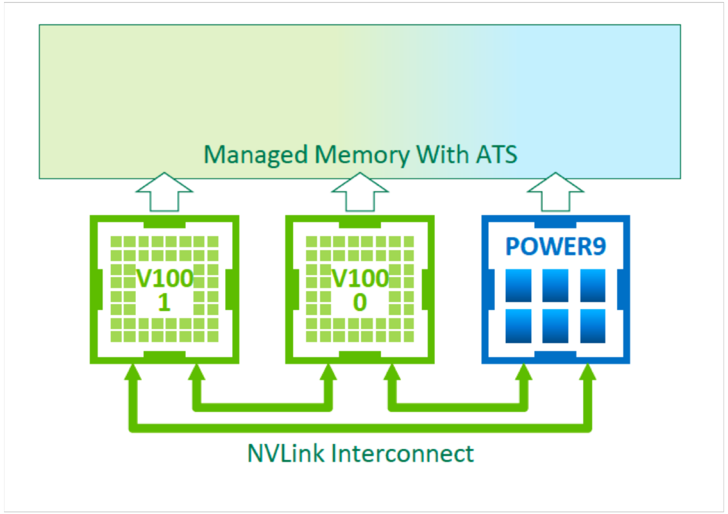
IBM AC922 Deep Learning System Architecture

AC922-GTW



Unified Memory with ATS on IBM POWER9

IBM POWER9 CPUs With NVLink Interconnect



ALLOCATION

- Automatic access to all system memory: malloc, stack, file system

ACCESS

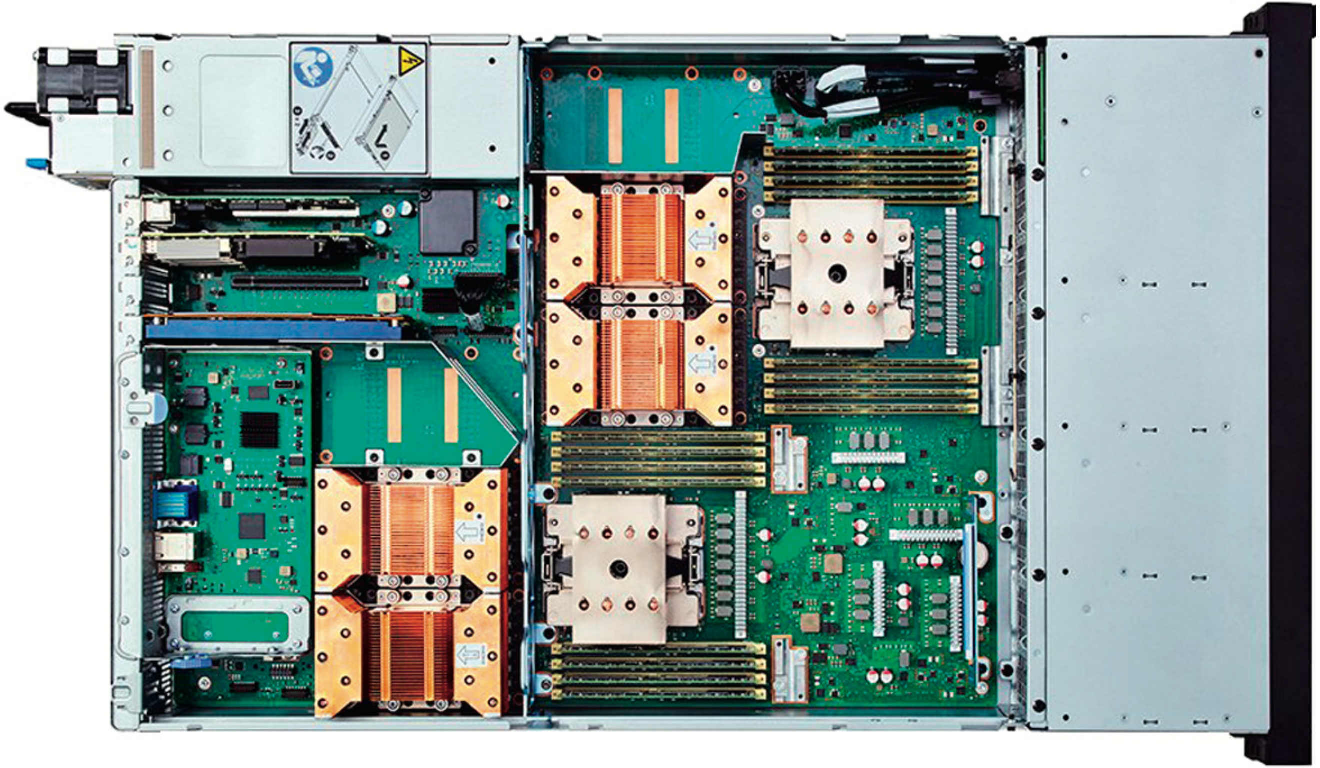
- All data accessible concurrently from any processor, anytime
- Atomic operations resolved directly over NVLink

ATS & POWER9 FEATURES

- ATS allows GPUDirect RDMA to unified memory
- Managed memory is cache-coherent between CPU and GPU
- CPU has direct access to GPU memory without need for migration

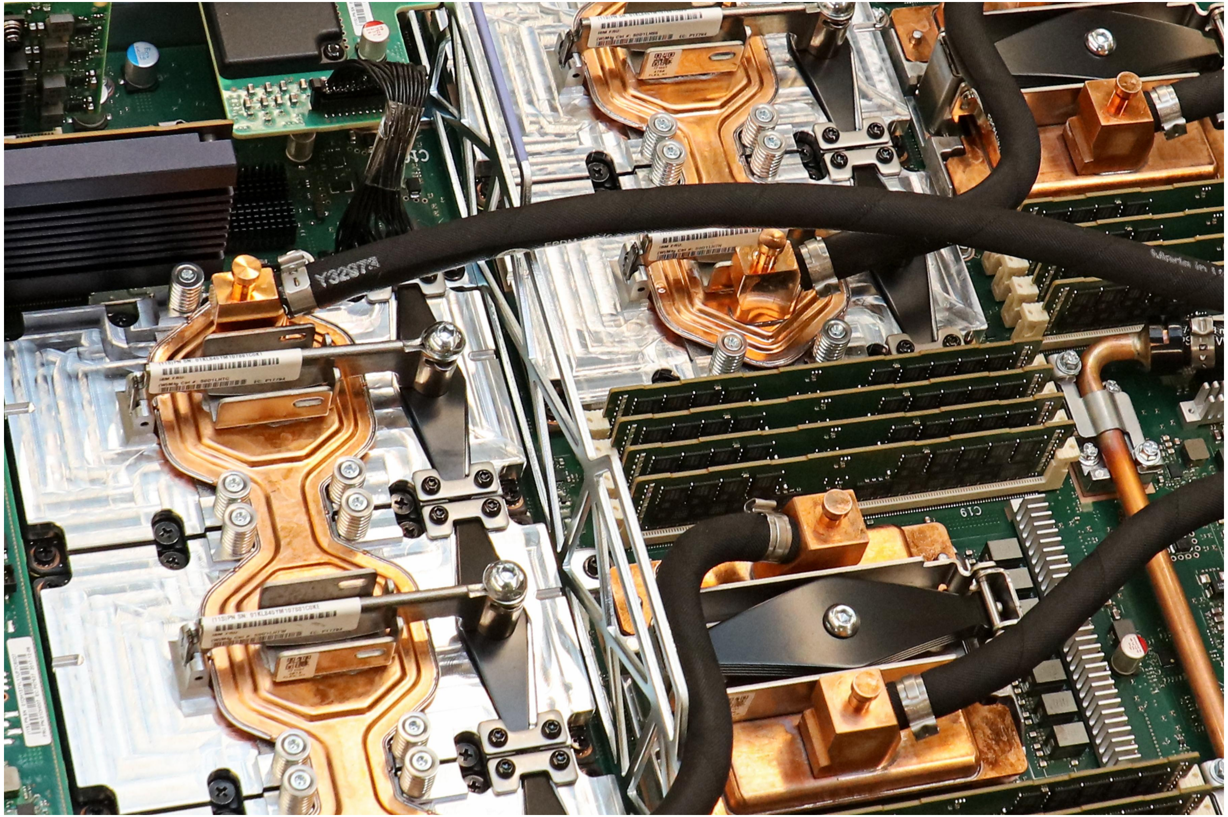
IBM AC922 Deep Learning System

AC922-GTG



IBM AC922 Deep Learning System

AC922-GTW



IBM AC922 System

Options and Features

Processor Features

- **16 Core** Processor Module
  190W – 250W (2.25GHZ - 3.12GHZ)
- **20 Core** Processor Module
  190W – 250W (2.25GHZ - 2.80GHZ)

- **18 Core** Processor Module
  190W – 250W (2.98GHZ - 3.26GHZ)
- **22 Core** Processor Module
  190W – 250W (2.78GHZ - 3.07GHZ)

Memory Features

- 8GB IS RDIMM DDR4
- 16GB IS RDIMM DDR4
- 32GB IS RDIMM DDR4
- 64GB IS RDIMM DDR4
- 128GB IS RDIMM DDR4

Storage Features

- HDD 1TB 2.5” 7k RPM SATA
- HDD 2TB 2.5” 7k RPM SATA
- SSD 960GB 2.5” SATA
- SSD 1.92TB 2.5” SATA
- SSD 3.84TB 2.5” SATA

- 1.6TB NVMe Flash Adapter
- 3.2TB NVMe Flash Adapter
- 6.4TB NVMe Flash Adapter

OpenPower Recent Tests on PCIe Gen4

- **GEN4 X16 NETWORK CARD**
  Mellanox CX5 Mezz 2.0
  - *Gen3 Limitation: ~94 Gb/s*

Result: **187.7 Gb/s**

```
[  3] local 10.0.0.22 port 44573 connected with 10.0.0.21 port 5001
[  3] local 10.0.0.24 port 38176 connected with 10.0.0.23 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  108.6 GBytes  93.6 Gbits/sec
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  109.2 GBytes  94.1 Gbits/sec
```

GEN4 X8 STORAGE ADAPTER
Eideticom NVM Express Offload

Gen3 Limitation: ~6.8 GB/s

Result: **13.5 GB/s**

```
ubuntu@ubuntu:~/NoLoad-Demos/fio$ sudo fio config.fio --filename=/dev/nvme0n1
simple: (g=0): rw=read, bs=4M-4M/4M-4M/4M-4M, ioengine=libaio, iodepth=32
...
fio-2.2.10
Starting 16 processes
^Cbs: 16 (f=16): [R(16)] [45.2% done] [13560MB/0KB/0KB /s] [3390/0/0 iops] [e
fio: terminating on signal 2

simple: (groupid=0, jobs=16): err= 0: pid=2626: Wed Feb 21 14:37:50 2018
  read : io=178768MB, bw=13551MB/s, iops=3387, runt= 13192msec
```

IBM AC922 System

Options and Features

PCIe Adapter Features

- 4-Port Ethernet (4x1 1Gb)
- 2-Port 40/100 GbE RoCE SFP+
- 2-Port Ethernet (10Gb)
- 4-Port Ethernet (2x10 10Gb Optical + 2x 1Gb)
- 4-Port Ethernet Cu (2x10 10Gb CU + 2x 1Gb)
- 2 Port 10Gb/s NIC & ROCE SR/CU
- 2 Port 25/10Gb/s NIC & ROCE SR/CU

- 1 Port EDR 100Gb IB CX-5 CAPI
- 2 Port EDR 100Gb IB CX-5 CAPI

- 2-Port Fiber Channel (16Gb/s)
- 2-Port Fiber Channel (32Gb/s)

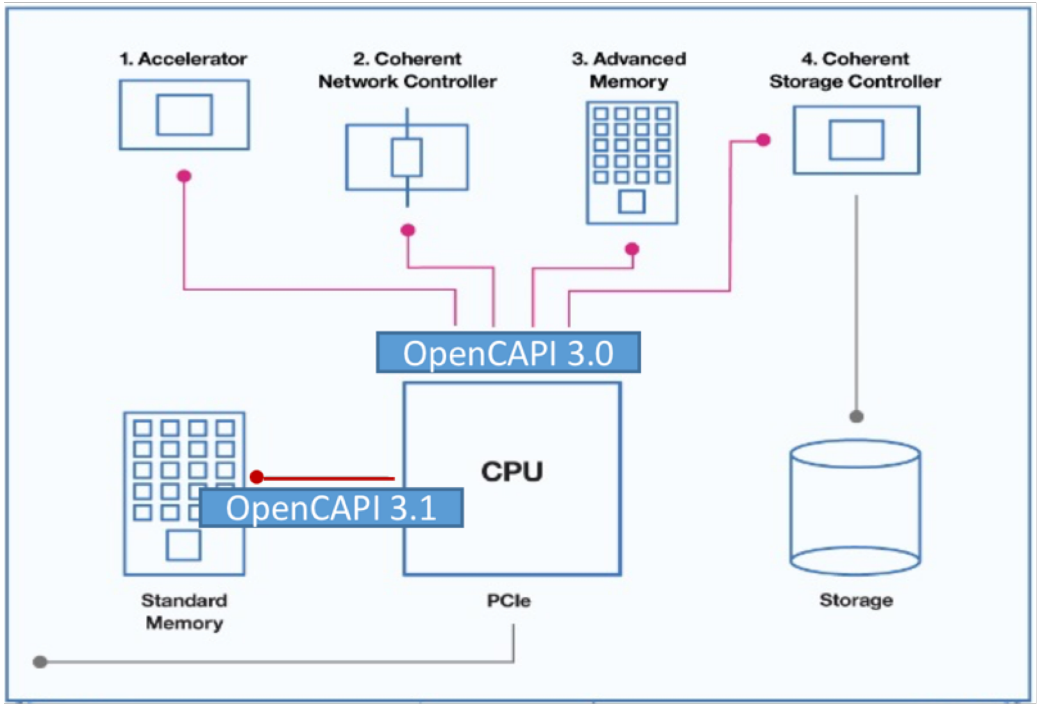
Accelerators Features

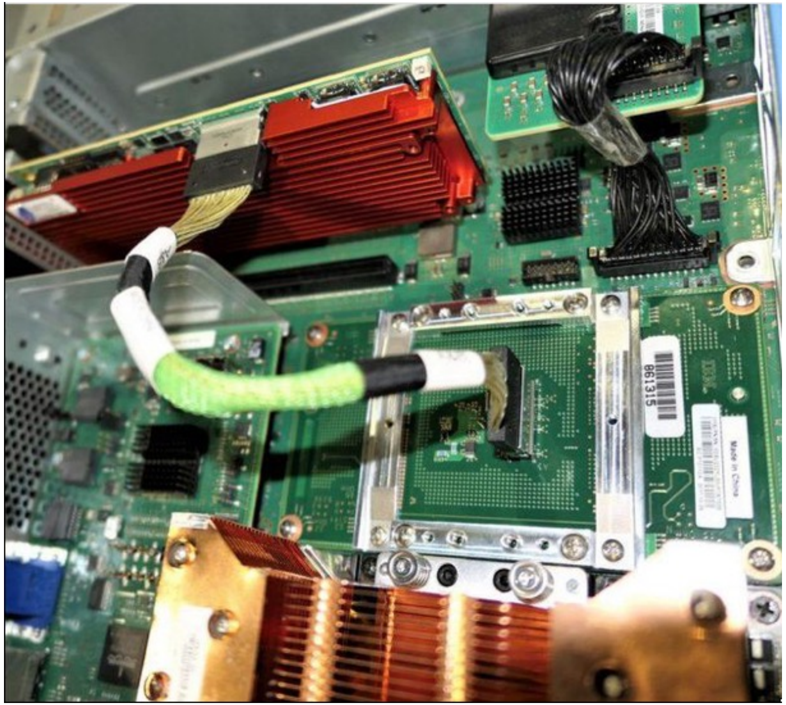
- NVIDIA V100 SMX2 16GB HBM2
- NVIDIA V100 SMX2 32GB HBM2

- Xilinix ADM-PCIE-8V3 FPGA

OpenCAPI 3.0

Data-Centric approach to server design





IBM AC922 Deep Learning System

Front and Rear View

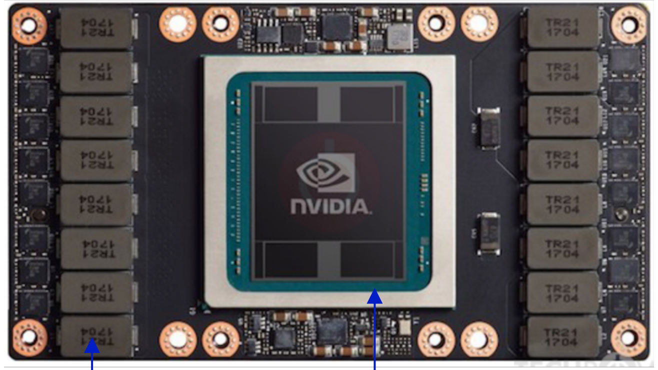
Front View

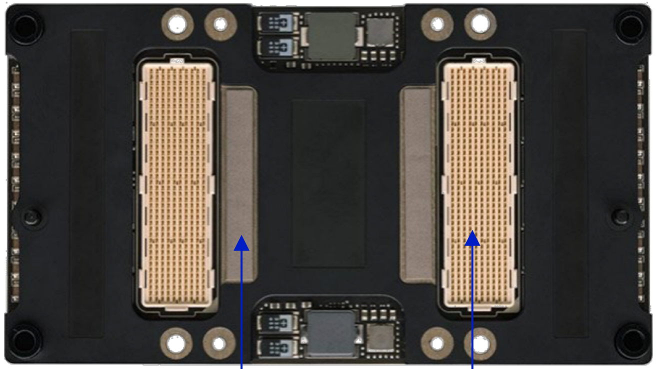
Rear View

NVIDIA GPU Details

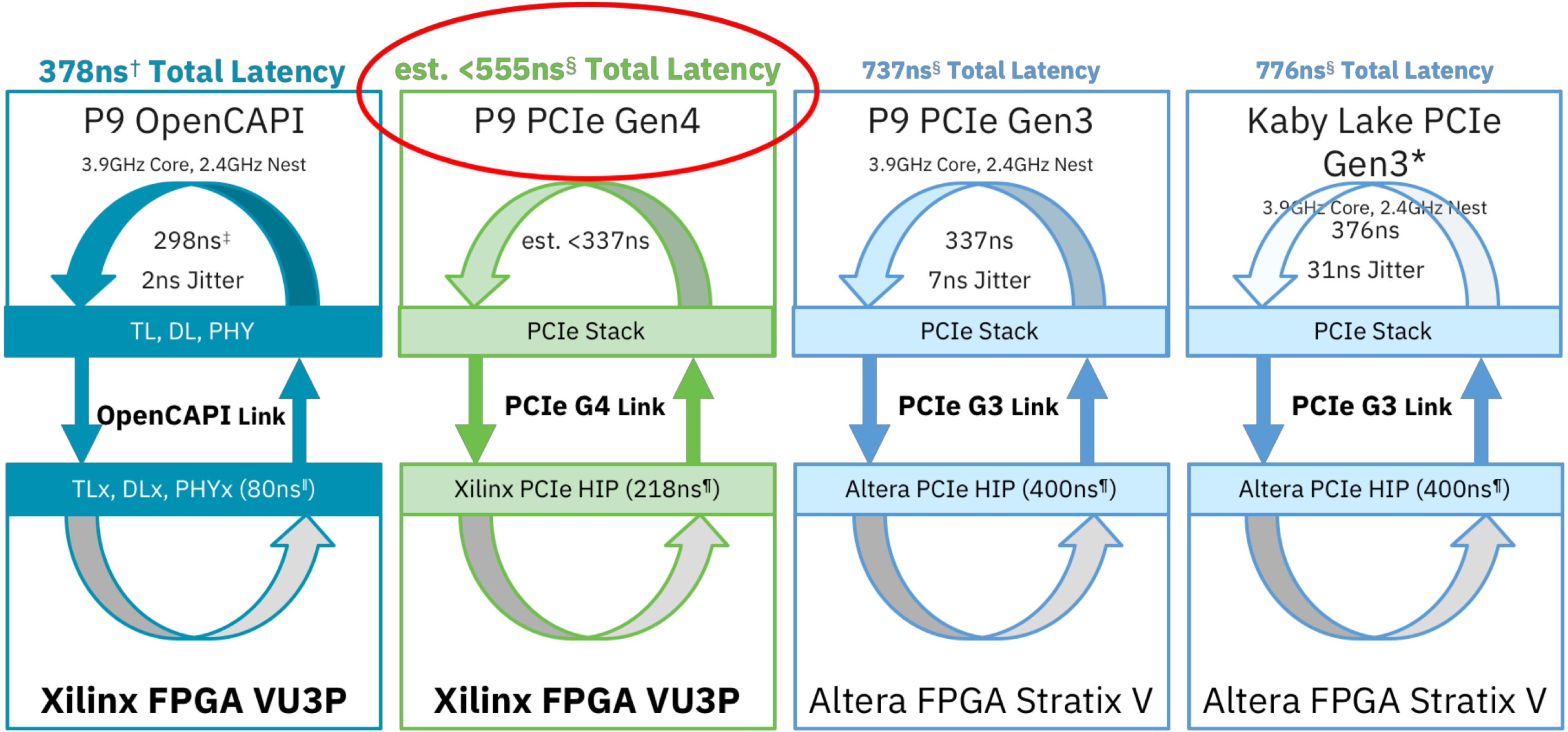
Volta SMX2 GPU Accelerator

NVIDIA Volta Specifications	
NVIDIA Tensor Cores	640
NVIDIA CUDA Cores	5120
Peak Double-Precision Performance	7.8 TFLOPS
Single-Precision Performance	15.7 TFLOPS
Tensor Performance	125 TFLOPS
Memory Bandwidth	900 GB/sec
GPU Memory Size	16 GB or 32GB HBM2
NVLink “Bricks” (8 lane interface)	6
NVLink Interconnect Bi-Directional	300 GB/sec
Maximum Power	300W





CAPI Advantages on AC922 Deep Learning System



* Intel Core i7 7700 Quad-Core 3.6GHz (4.2GHz Turbo Boost)
† Derived from round-trip time minus simulated FPGA app time
‡ Derived from round-trip time minus simulated FPGA app time and simulated FPGA TLx/DLx/PHYx time

§ Derived from measured CPU turnaround time plus vendor provided HIP latency
ǁ Derived from simulation
¶ Vendor provided latency statistic

OpenBMC is a free open source management software Linux distribution

IBM is the OpenBMC Community Leader

- Facebook
- Google
- IBM
- Intel
- Microsoft
- OCP

Feature List:

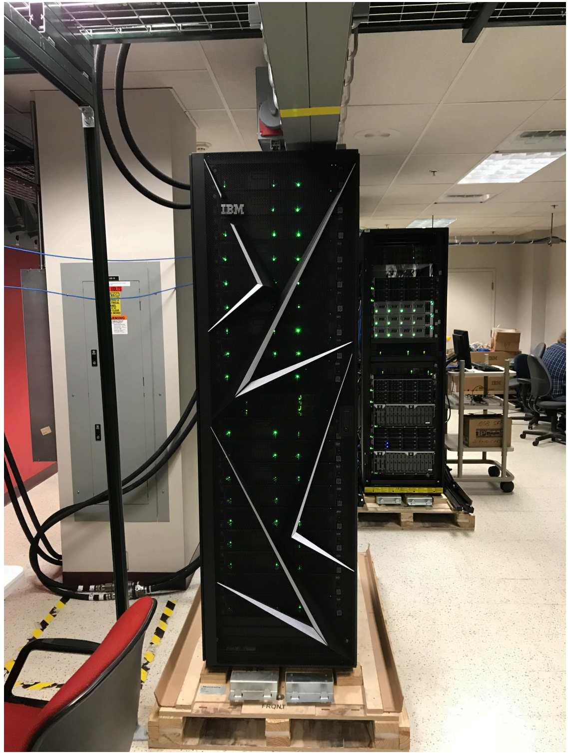
- REST Management
- IPMI
- SSH based SOL
- Power and Cooling Management
- Event Logs
- Zeroconf discoverable
- Sensors
- Inventory
- LED Management
- Host Watchdog
- Simulation
- Code Update Support for multiple BMC/BIOS images
- POWER On Chip Controller (OCC) Support

Features In Progress:

- Full IPMI 2.0 Compliance with DCMI
- Verified Boot
- HTML5 Java Script Web User Interface
- BMC RAS

IBM AC922 Deep Learning System Cluster POD

72x or 108x NVIDIA Volta V100 GPU's Example



IBM Deep Learning Software Stack

PowerAI family

AI for Data Scientists and non-Data Scientists

PowerAI Vision
Auto-DL for Images & Video

- Label
- Train
- Deploy

Driverless AI
Auto-ML for Text & Numeric Data, NLP

- Import
- Experiment
- Deploy

PowerAI

PowerAI: Open Source ML Frameworks
TensorFlow™ Caffe PyTorch

- Large Model Support (LMS)
- Distributed Deep Learning (up to 4 nodes)

WML Accelerator (formerly PowerAI Enterprise)

- Distributed Deep Learning (DDL – 1000s of nodes)
- Auto Hyper-parameter Tuning

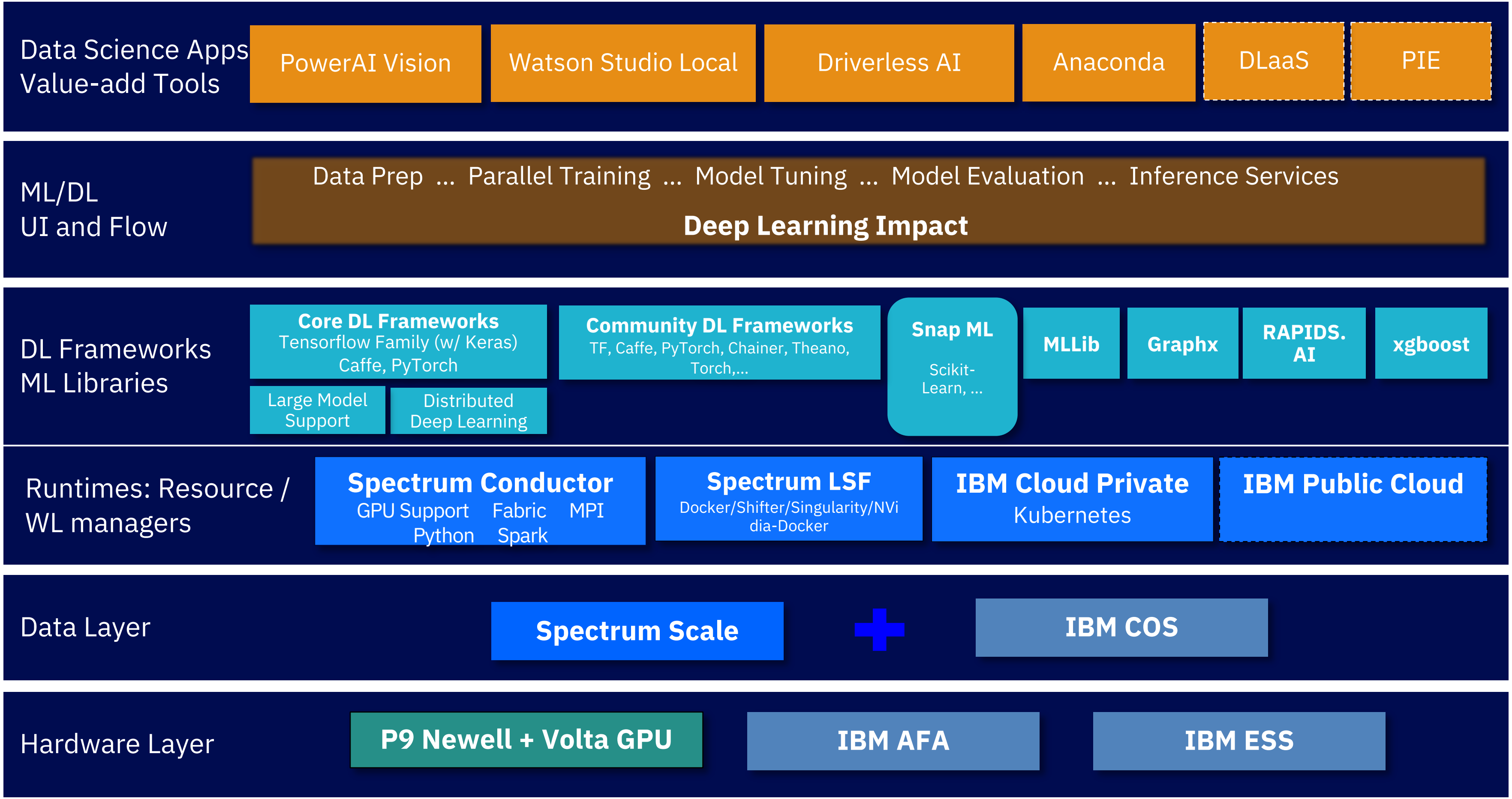
IBM Spectrum Conductor with Spark
Cluster Virtualization,
Dynamic Resource Orchestration,
Multiple Frameworks, Distributed Execution Engine

Deep Learning Impact (DLI) Module

Data & Model Management, ETL, Visualize, Advise

Accelerated Infrastructure

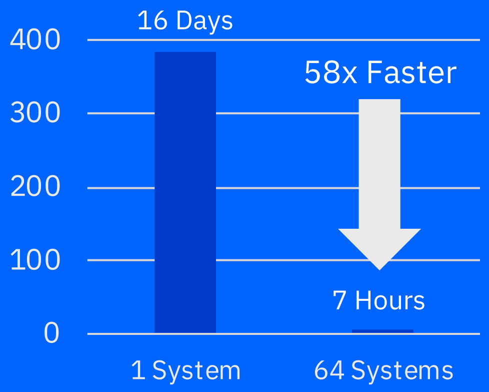
Reference Architecture for AI Infrastructure: Software



Research Innovations

Optimized ML/DL frameworks & libraries

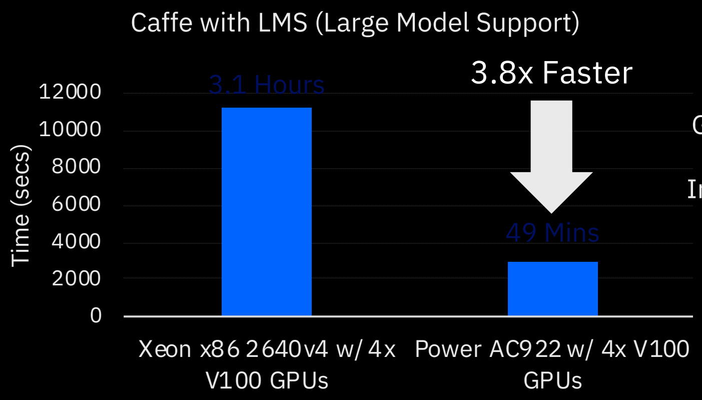
Distributed Deep Learning



Caffe with PowerAI DDL,
Running on Minsky (S822Lc)
Power System

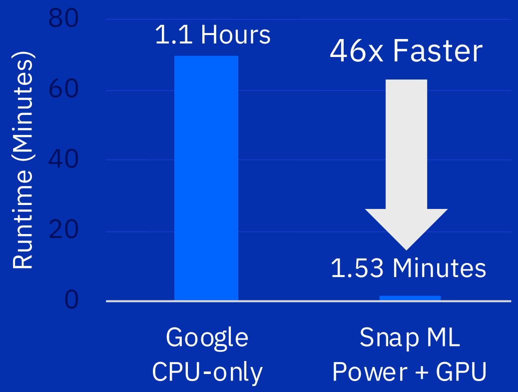
ResNet-101, ImageNet-22K

Large Model Support



GoogleNet model on Enlarged ImageNet Dataset (2240x2240)

Snap Machine Learning



Logistic Regression in Snap ML (with GPUs) vs TensorFlow (CPU-only)

Google: 90 x86 servers
Snap ML: 4 AC922 servers

PowerAI 1.6.0

PowerAI packaging with CONDA

PowerAI

Channel

CONDA

jupyter, spyder, NumPy, SciPy, Numba

pandas, DASK, Bokeh, HoloViews, Datashader

matplotlib, scikit-learn, H2O.ai, TensorFlow, CONDA

What is Conda
(and why should I care?)

- Anaconda started as an open-source distribution of Python.
  - But it’s also expanded to cover other languages and software
  - Easy access to thousands of packages
    - “conda install” for pre-compiled packages
    - pip still available
- Provides virtual environments for non-Python software
- Puts software management in hands of the user
  - No waiting for consultants to install packages
  - Control over which versions of packages are installed
  - Provides some package management routes for other languages

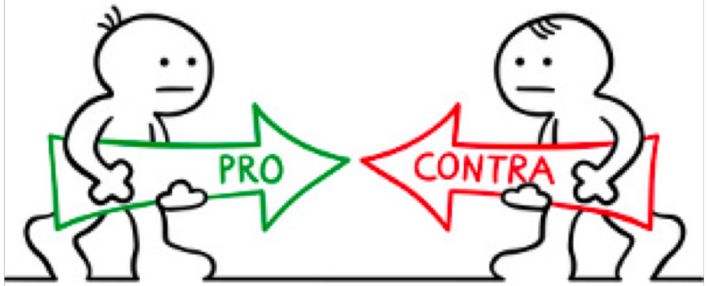
Pros and Cons of Adopting Anaconda

Advantages:

Easy access to a broad variety of software and their dependencies
Portability of environments
Opportunity to revisit some stale workflows
Users have control of the software

Disadvantages

Users have control of the software
Some duplication of packages in user environments



PowerAI
Delivery options

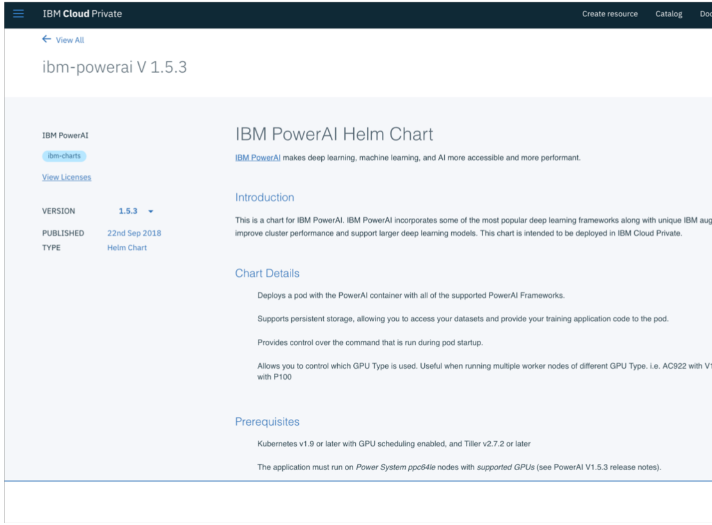
PowerAI

PowerAI

ubuntu

ubuntu redhat

docker



PowerAI

Version 1.6.0

TensorFlow

Estimator, Probability, TF.Keras, Tensorboard

DDL-TensorFlow

Bazel

Caffe

Py-OpenCV

PyTorch

OpenCV

cuDF

cuML

SnapML

Local, MPI, Spark

MAGMA

Distributed Deep Learning (DDL)

Large Model Support (LMS)

OpenMPI

ONNX

LIBS:

NCCL

cuDNN

libevent, libgdf, libgdf_cffi, libopencv, libprotobuf, parquet-cpp, thrift-cpp, arrow-cpp, pyarrow, gflags etc

CUDA Toolkit

GPU Drivers

Ubuntu 18

Red Hat Linux 7

PowerAI
Docker Container

Component	1.5.2 Images	1.5.3 Images	1.5.4 Images	1.6.0 Images
Distributed Deep Learning (DDL)	1.0.0	1.1.0	1.2.0	1.3.0
TensorFlow	1.8.0	1.10.0	1.12.0	1.13.1
TensorFlow Probability	NA	NA	0.5.0	0.6.0
TensorFlow Estimator	NA	NA	NA	1.13.0
TensorBoard	1.8.0	1.10.0	1.12.0	1.13.0
IBM Caffe	1.0.0	1.0.0	1.0.0	1.0.0
BVLC Caffe	1.0.0	1.0.0	1.0.0	NA
Caffe2	NA	NA	1.0rc1	1.0.1
PyTorch	0.4.0	0.4.1	1.0rc1	1.0.1
Snap ML	1.0.0	1.0.0	1.0.0	1.2.0
Spectrum MPI	10.2	10.2	10.2	10.2
Bazel	0.10.0	0.15.0	0.15.0	0.20.0
OpenBLAS	0.2.20	0.3.2	0.3.3	0.2.20
Protobuf	3.4.0	3.4.0	3.6.1	3.6.1
ONNX	NA	NA	1.3.0	1.3.0
Rapids cuDF	NA	NA	NA	0.2.0
Rapids cuML	NA	NA	NA	0.2.0

PowerAI
New Containers with Individual Frameworks

- Base repository image (no frameworks installed)
- Tensorflow based image (py2, py3)
- Pytorch based image (py2, py3)
- Caffe-ibm based image (py2, py3)
- SnapML based image (py2)
- All frameworks (py2, py3)

More choice, more flexibility, more simplicity

PowerAI
Large Model Support (LMS)

Problem

- Datasets are **large** and growing
- The size of a batch of samples is **large** and growing
- Sample sizes are **large** and growing
- More and more sophisticated models are being designed, some with hundreds of layers
- GPU memory capacity is growing as well (but slower)
- Limited by cost, technology, physical space

Problem

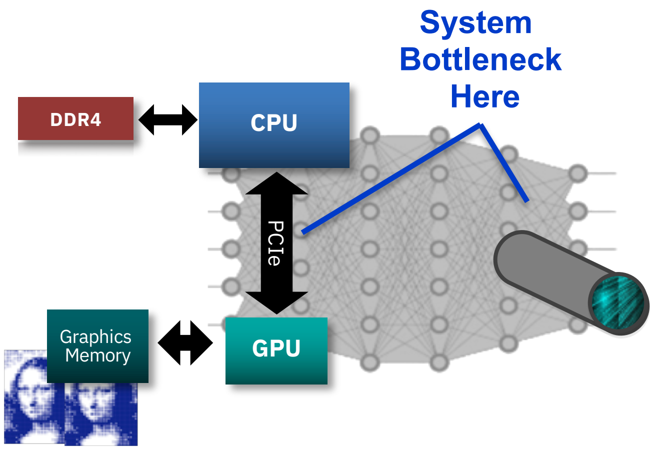
- So stay within the bounds then?

Well..
We don't like constraints!
We've already paid for memory in this system! Why can't we use that?
I'm using a batch size of 1 and am already pushing the limits, I can't compromise any more!

Train Larger More Complex Models

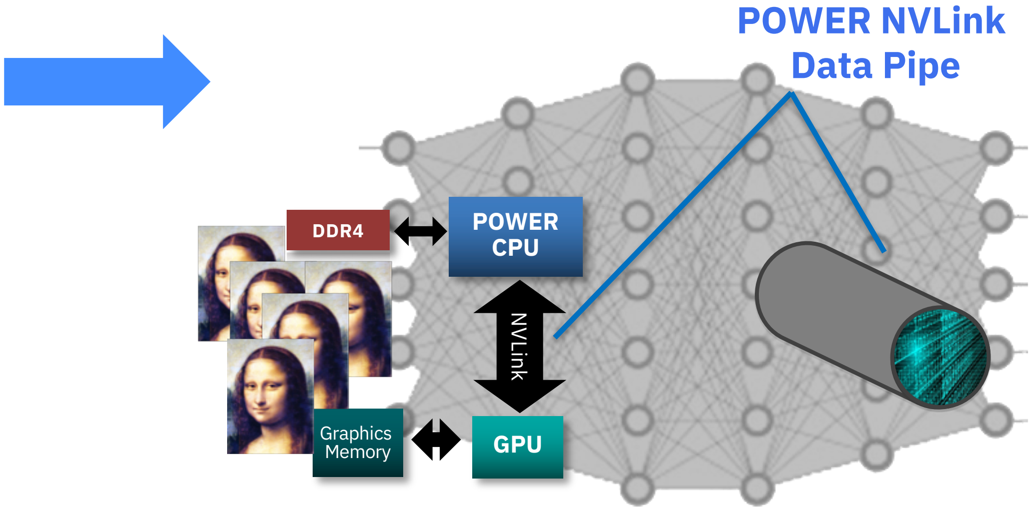
Traditional Model Support

Limited memory on GPU forces tradeoff in model size / data resolution



Large Model Support

Use system memory and GPU to support more complex and higher resolution data



Large Model Support gets Bigger

- Introduction of TensorFlow Large Model Support v2
  ... increases image size resolution
  ... improved performance for complex networks
  ... this version is closed source for now
- With Large Model Support v1 (open source)
  ... TensorFlow (separate package from TF w/ LMSv2)
  ... Caffe
  ... PyTorch

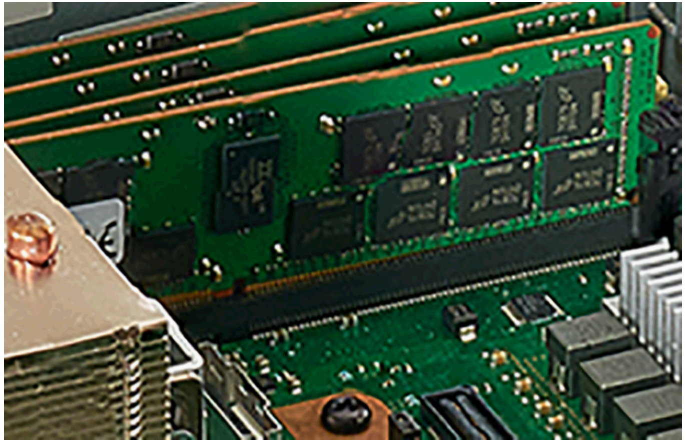
  Large Model Support v1 and v2 is now fully supported
  in PowerAI: no longer technical preview

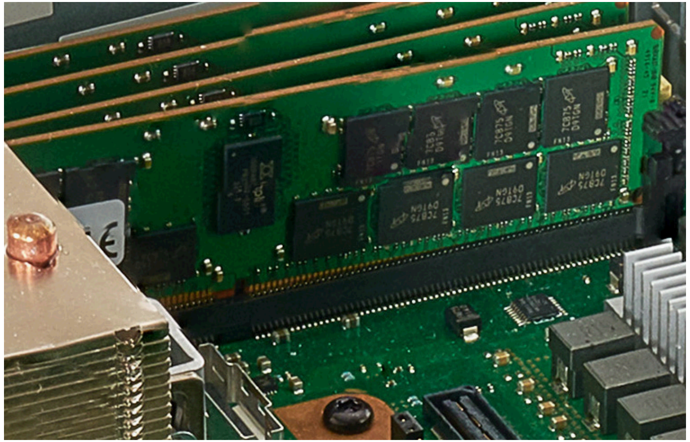
What’s possible with TFLMS v2

5x MRI resolution – 3D U-Net

- 304^3 w/16 GB GPU
- 400^3 w/ 32 GB GPU

10x image resolution - ResNet50 and DeepLabV3 2D image segmentation





Easier to enable in model code

Keras API

```
from tensorflow_large_model_support import LMS
lms = LMS()
lms.batch_size = 1
# ...
model.fit_generator(generator=training_gen,
                    callbacks=[lms])
```

Estimator API

```
from tensorflow_large_model_support import LMS
lms = LMS()
# ...
mnist_classifier.train(input_fn=train_input_fn, steps=20000,
                       hooks=[logging_hook, lms])
```

LMS Usage in IBM-Caffe

LMS enables processing of high definition images, large models, and higher batch sizes that doesn’t fit in GPU memory today (Maximum GPU memory available in Nvidia P100 GPUs is 16GB).

LMS Options

- lms <size in KB>
- lms_frac <x>, where 0<x<1.0

Example of running IBM Caffe with LMS for Deep Residual Network – Resnet152 :
/opt/DL/caffe-ibm/bin/caffe train -gpu 0,1,2,3 –solver=solver.prototxt -lms 10000 –lms_frac=0.5

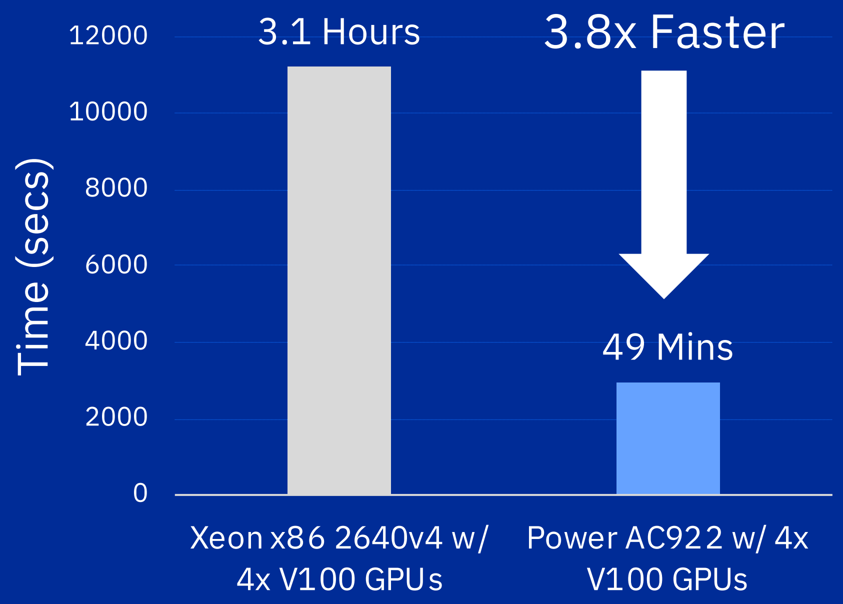
Note that configuring the “lms” and “lms_frac” values depends on the below factors:

- Batch size used
- Model used
- Number of GPUs used
- System memory available

Large AI Models Train ~4 Times Faster

POWER9 Servers with NVLink to GPUs
vs
x86 Servers with PCIe to GPUs

Caffe with LMS (Large Model Support)
Runtime of 1000 Iterations



GoogleNet model on Enlarged ImageNet Dataset (2240x2240)

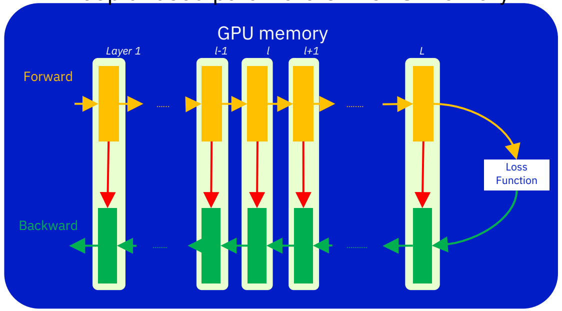
LMS in TensorFlow

Enabling large models and datasets

- TFLMS modifies the TensorFlow graph prior to training to inject swap nodes that will swap tensors in and out of GPU memory to system memory.
- Contributed to the community
  https://github.com/tensorflow/tensorflow/pull/19845
- Large bandwidth of NVLink2 makes this perform well while enabling the graph to train against larger datasets, higher resolutions and/or large models.
- Relies on an existing contrib module, tf.contrib.graph_editor
  https://www.tensorflow.org/api_docs/python/tf/contrib/graph_editor

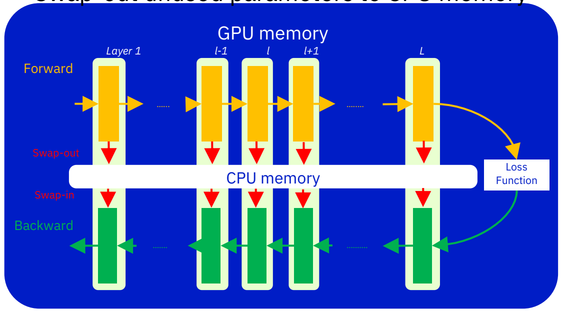
Normal Backpropagation

Keep unused parameters in GPU memory



Backpropagation in LMS (Swap)

Swap-out unused parameters to CPU memory



LMS Use Cases

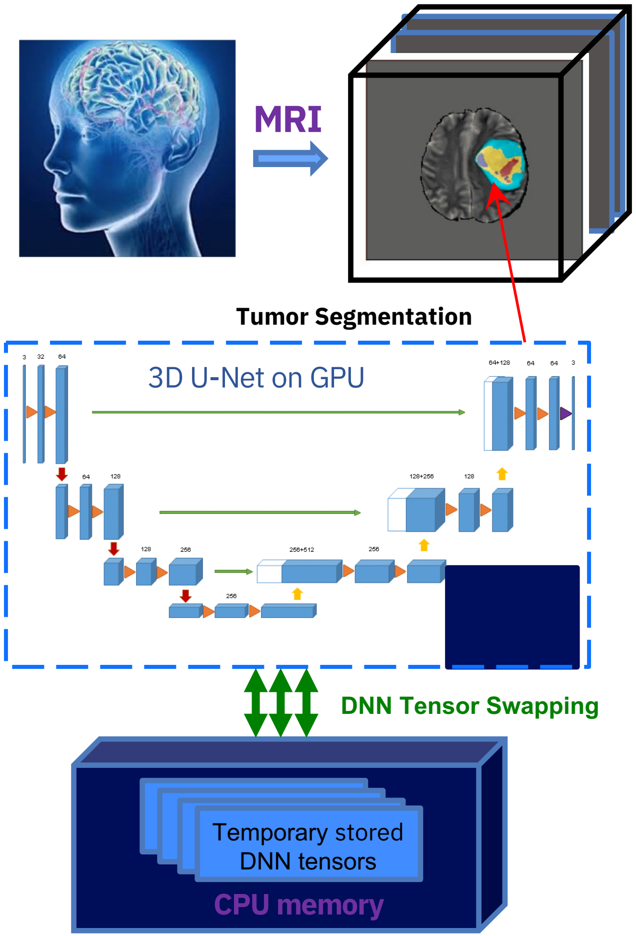
Segmentation of Brain tumor MRI

– GPU memory is too small

Large MRI scans cannot be processed by 3D-Unet even for single sample batch

– Typical solutions (Resolution reduction, Image partitioning) result in accuracy loss

Comparison / Projection		
Use Case	Power AC922	x86
MRI Segmentation	4 images per computation	1 image per computation
Resnet50	250 images/sec	65 images/sec



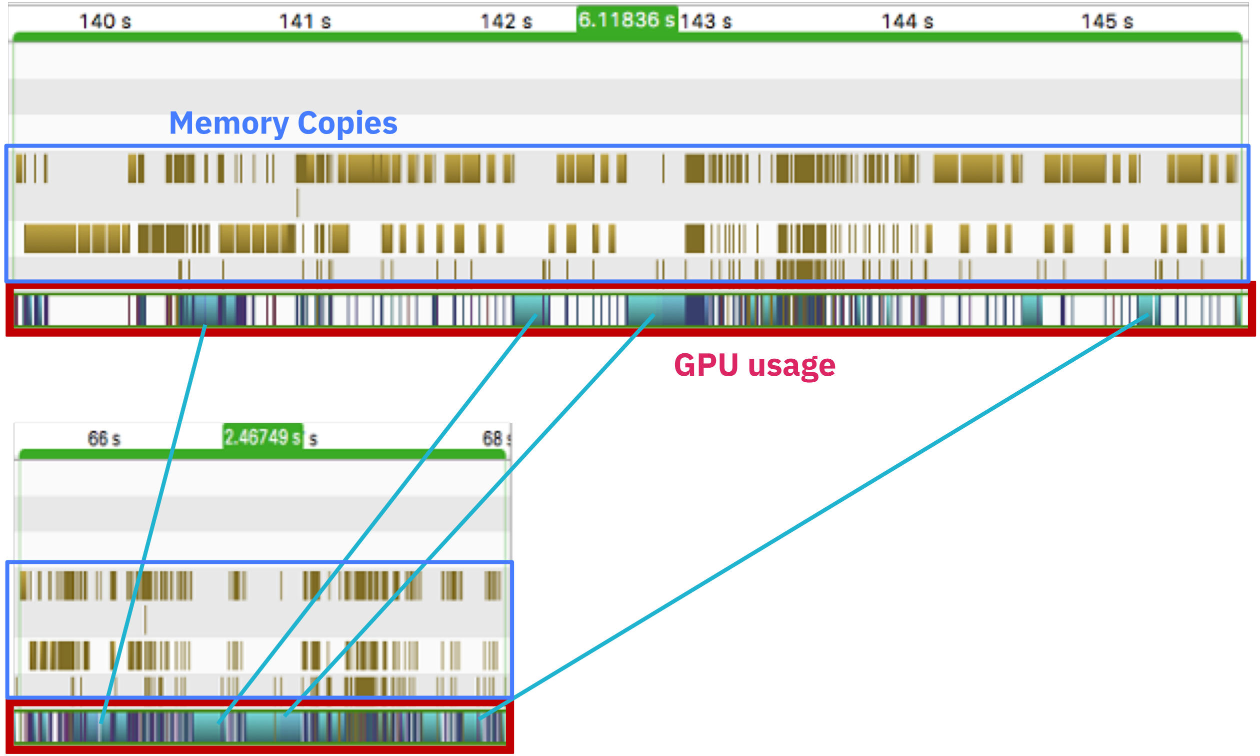
TFLMS profiling: NVLink2 advantage over PCIe

Training 3DUnet models for image segmentation has high memory usage requirements which can limit the size of the 3D images used for training

TFLMS enables larger models and images by swapping tensors between GPU and System memory

The high BW interconnects between GPUs, CPUs and System Memory of the AC922 enables the Large Model Support

PCIe connected GPU training one high res 3D MRI with large model support



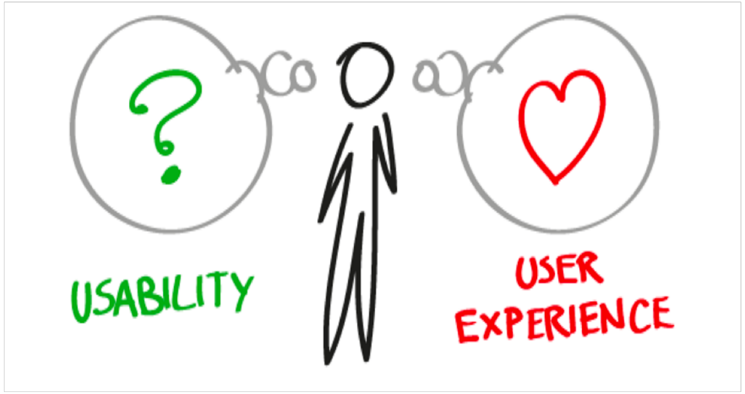
NVLink 2.0 connected GPU training one high res 3D MRI with large model support

PowerAI
Distributed Deep Learning (DDL)

Distributed Deep Learning
Goals

The overall goal of distributed deep learning is to reduce the training time

To this end the primary features:
- Automatic Topology Detection
- Rankfile generation
- Automatic mpirun option handling
- Efficiency in scalability



Distributed Deep Learning

How is working?

- A process is created for each GPU in the cluster
- Each process contains a copy of the model
- Mini-batch is spread across all of the processes
- Each process uses different input data
- After each iteration, all of the processes sync and average together their gradients

PowerAI

Distributed Deep Learning Library (DDL)

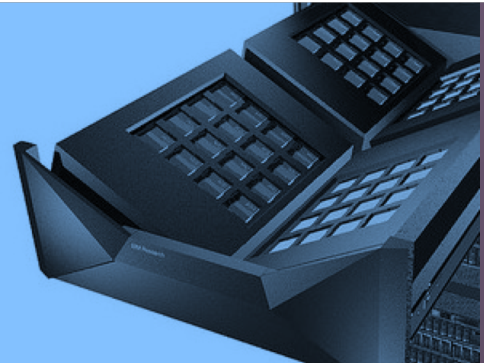
Communication Library for Distributed Deep Learning Training

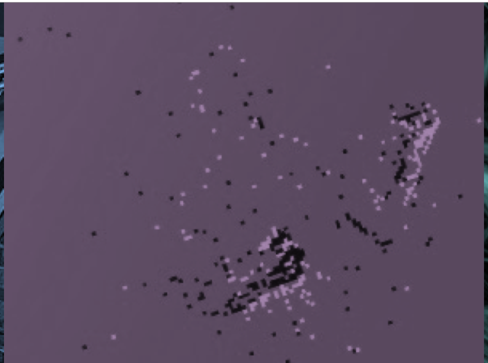
- Enables deep learning software to scale to 100s of servers with GPUs
- Works across variety of system sizes
- Works with variety of network types, switch topologies

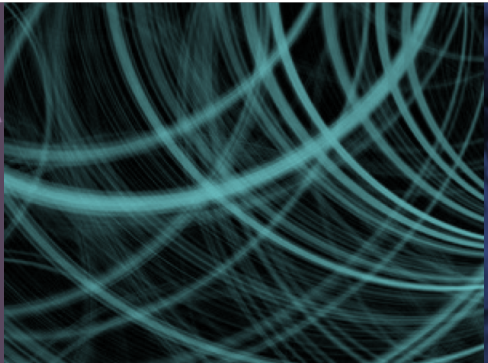
Released results @ 256 P100 GPUs

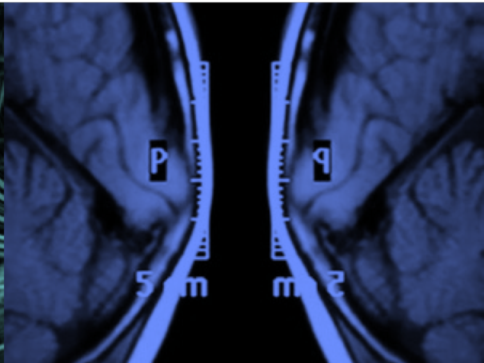
- Better scaling efficiency than Facebook AI Research: 95% (IBM) vs <90% (FB)
- Higher image recognition accuracy than Microsoft: 33.8% (IBM) vs 29.8% (MS)

TECHNICAL DETAILS:
https://arxiv.org/abs/1708.02188









Distributed Deep Learning (DDL)

Deep learning training takes days to weeks

Limited scaling to multiple x86 servers

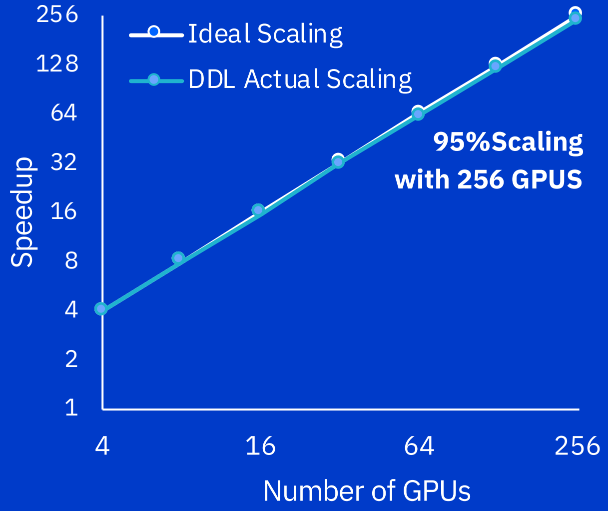
PowerAI with DDL enables scaling to 100s of GPUs

16 Days Down to 7 Hours

58x Faster

ResNet-101, ImageNet-22K

Near Ideal Scaling to 256 GPUs



ResNet-50, ImageNet-1K

Caffe with PowerAI DDL, Running on Minsky (S822Lc) Power System

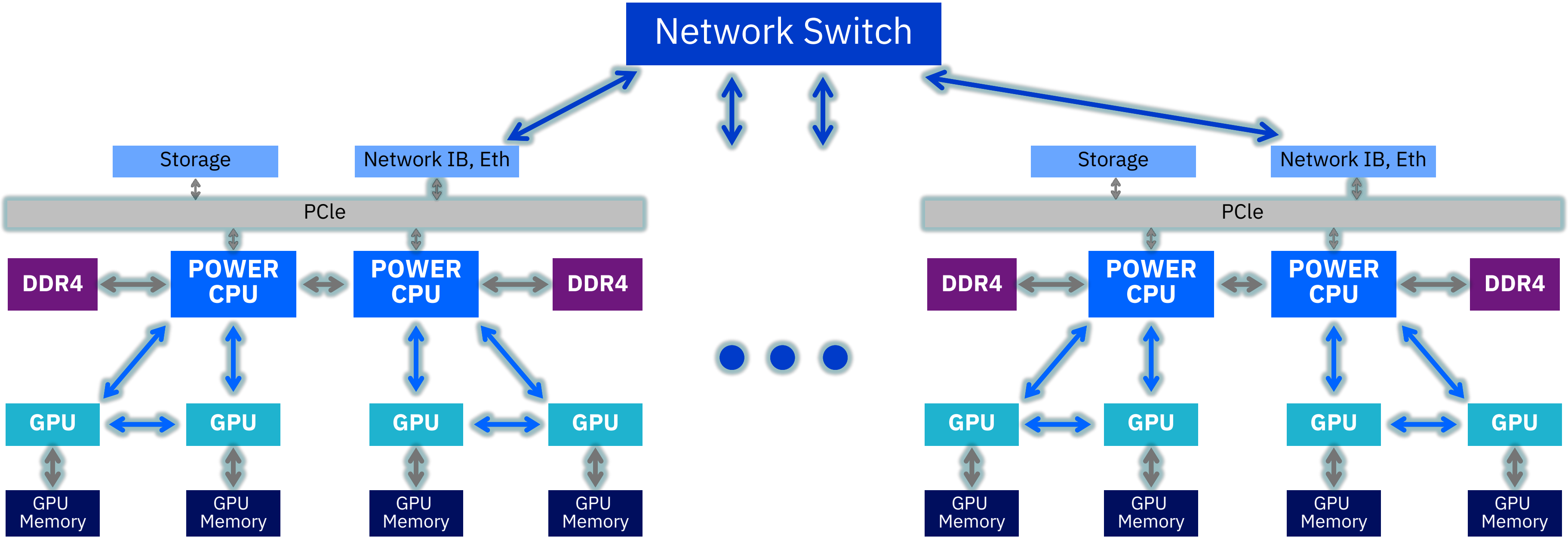
What does DDL do?

DDL for TensorFlow	
	1. Places the job on the local GPU to the CPU (negotiating to use NVLink interface)
	2. Places the job on its nearest neighbor, to leverage NVLink GPU:GPU communication
	3. Places the job on the same system, on the other socket
	4. Sends the job, integrating RDMA over IB (not present in the frameworks themselves), to a remote system and it's first GPU

Same kind of intelligence you see in good HPC job schedulers, but created with specific tuning for our architecture

PowerAI DDL Dimensions

Communication paths



DDL splits reductions into different dimensions, using different algorithms for each dimension.

PowerAI DDL

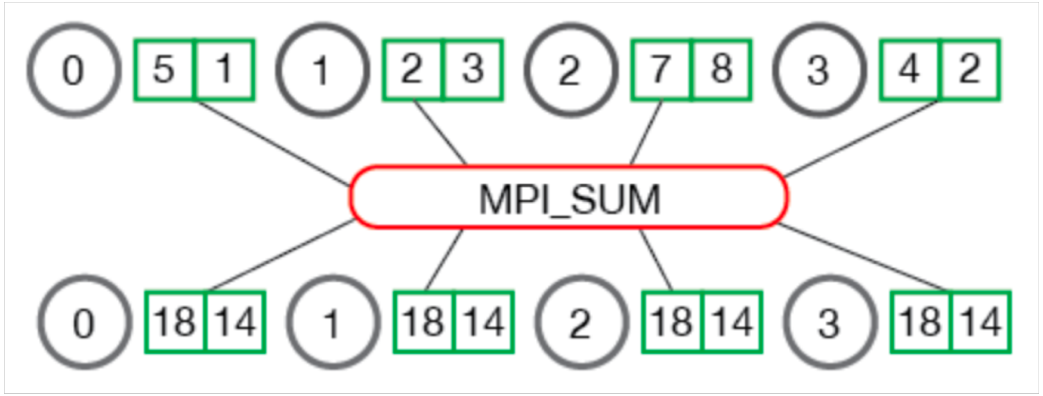
IBM Distributed Deep Learning Library provides:

- A C library that provides functions needed to perform distributed deep learning operations (such as allreduce)
- The library utilizes the MPI and NCCL libraries
- A tool for launching jobs across a cluster called **ddlrun**
- Framework integrations:
  - Provides a custom operator for TensorFlow, plus python wrappers around DDL library
  - DDL integration is built into Caffe and PyTorch

PowerAI DDL

allReduce

- allReduce performs an element-wise reduction on arrays of data spread across nodes of a cluster
- At the end of the allReduce calculation, every node will have a copy of the result
- DDL provides support for the **sum** and **average** reduction operations



PowerAI DDL

Run

Launch a program using ddlrun:
ddlrun --H host1,host2 caffe train --solver=SOLVER.prototxt

Launch a program using ddlrun:
ddlrun --H host1,host2 python MY_SCRIPT.py

Common ddlrun arguments:

» --m : Select the DDL mode
» --accelerators : Specify the number of GPUs per node to use
» --tcp : Use TCP communication between nodes instead of Infiniband
» --mpiarg : Pass along extra MPI arguments
» --verbose : Provides extra output describing checks that are performed
» --skipchecks : Don't perform network checks

PowerAI DDL Modes

Some of the available modes are:

- b : Uses lower level NCCL functions. This generally gets the best performance between GPUs in the same node.
- n : Uses higher level NCCL functions.
- r : Performs a ring based reduction using MPI commands.
- m : Uses higher level MPI functions. This can be used on clusters without GPUs.
- p : Determines the best mode to use for each dimension. There is a small startup cost and larger upfront GPU memory usage when using p mode.

There are several different reduction algorithms that DDL implements (called modes).

The user can choose which mode to use for each dimension of the calculation

PowerAI DDL

Automatic Topology Detection and Rankfile generation

Another common source of frustration when getting started with DDL is the generation of the rankfile.
With the version from PowerAI 1.5.2 of ddlrun, the topology is inferred from the host list and a rankfile is automatically generated by discovering the configuration of the first host in the host list and verifying that all other hosts have the same configuration.

```
$ ddlrun -H host1,host2,host3,host4 python .....
```

This command will automatically generate and use the following rankfile:

```
#host = host1,host2,host3,host4
#aisles = 1
#racks = 1
#nodes = 4
#accelerators = 4
#sockets = 2
#cores = 16

rank 0=host1 slot=0:0-7
rank 4=host1 slot=0:8-15
rank 8=host1 slot=1:0-7
rank 12=host1 slot=1:8-15

rank 1=host2 slot=0:0-7
rank 5=host2 slot=0:8-15
rank 9=host2 slot=1:0-7
rank 13=host2 slot=1:8-15

rank 2=host3 slot=0:0-7
rank 6=host3 slot=0:8-15
rank 10=host3 slot=1:0-7
rank 14=host3 slot=1:8-15

rank 3=host4 slot=0:0-7
rank 7=host4 slot=0:8-15
rank 11=host4 slot=1:0-7
rank 15=host4 slot=1:8-15
```

PowerAI DDL

Automatic mpirun option handling

There are quite a few options that have to be passed to mpirun every time a job is launched, and some that only need to be passed depending on what version of mpi is being used or how the environment is set up. ddlrun now handles these options automatically, displaying the fully constructed mpirun command it used. E.g.:

```
$ ddlrun -H host1,host2,host3,host4 python /mnist/mnist-env.py ...
+ mpirun -x PATH -x LD_LIBRARY_PATH -x DDL_OPTIONS -gpu --rankfile /tmp/ddlrun.BxI9Ufpz1Ycz/RANKFILE -n 16
python /mnist/mnist-env.py
```

If there's ever a need to pass additional options to mpirun, the --mpiarg option can be used. E.g.:

```
$ ddlrun --mpiarg "-pami_noib" -H host1,host2,host3,host4 python /mnist/mnist-env.py
```

PyTorch with DDL

- DDL is an option as a backend for PyTorch's Distributed class
  - For more information on the Distributed class, see: https://pytorch.org/docs/stable/distributed.html
- DDL enabled PyTorch scripts should be launched using ddlrun

PyTorch with DDL

- To use DDL, the ddl backend must be selected when calling init_process_group:

```
torch.distributed.init_process_group('ddl', init_method='env://')
```

- After that, PyTorch's DistributedSampler function and DistributedDataParallel functions can be used, and will use DDL to communicate between GPUs

```
kwargs = {'pin_memory': True}
train_sampler = torch.utils.data.distributed.DistributedSampler(dataset, world_size, rank)
train_data = torch.utils.data.DataLoader(dataset, batch_size=batch_size, shuffle=False,
                                         sampler=train_sampler, **kwargs)
```

```
model = torch.nn.parallel.DistributedDataParallel(model)
```

TensorFlow with DDL

PowerAI provides a custom TensorFlow operator for DDL.

This operator will allow the allReduce function to be called from within a TensorFlow script.

TensorFlow scripts will need to be modified to use DDL.

To use DDL with TensorFlow:

Launch a program using ddlrun:

– `ddlrun --H host1,host2 python MY_SCRIPT.py`

TensorFlow Distribution Strategy with DDL

If using TensorFlow's Estimators, DDL can be used without much modification to the script by using TensorFlow's Distribution Strategies

DDL works with the CollectiveAllReduceStrategy, which only works with Estimator's train_and_eval function

- The only modifications that need to be performed to an existing script that already uses Estimators train_and_eval function are:
  - Import the DDL library
  - Add the CollectiveAllReduceStrategy strategy to a TrainSpec

```
import ddl
```

```
#Define the distribution strategy
distribution_strategy = tf.contrib.distribute.CollectiveAllReduceStrategy(num_gpus_per_worker=1)

run_config = tf.estimator.RunConfig(train_distribute=distribution_strategy)
```

Modify TensorFlow Script to use DDL

If not using Distribution Strategies, the following changes must be made to a TensorFlow script to use DDL

– Import the DDL library

– Modify the batch size to be: Global Batch Size / Number of Workers

– Modify the learning rate to: Learning Rate * Number of Workers

– Modify script to only work on a certain section of the data, using Worker ID

– If using Keras

  - Add the *DDLCallback* and *DDLGlobalVariablesCallback* to the list of training call backs

– If using Estimator without Distribution Strategies

  - Add the *DDLGlobalVariableHook* to the list of training hooks

Splitting Data Between Processes

If training works by iterating over all of the data, each process should only iterate over a section of the data

- If using TFRecords, TF provides methods to help with this:
  - ds = tf.data.TFRecordDataset(filenames)
  - ds = ds.shard(ddl.size(), ddl.rank())

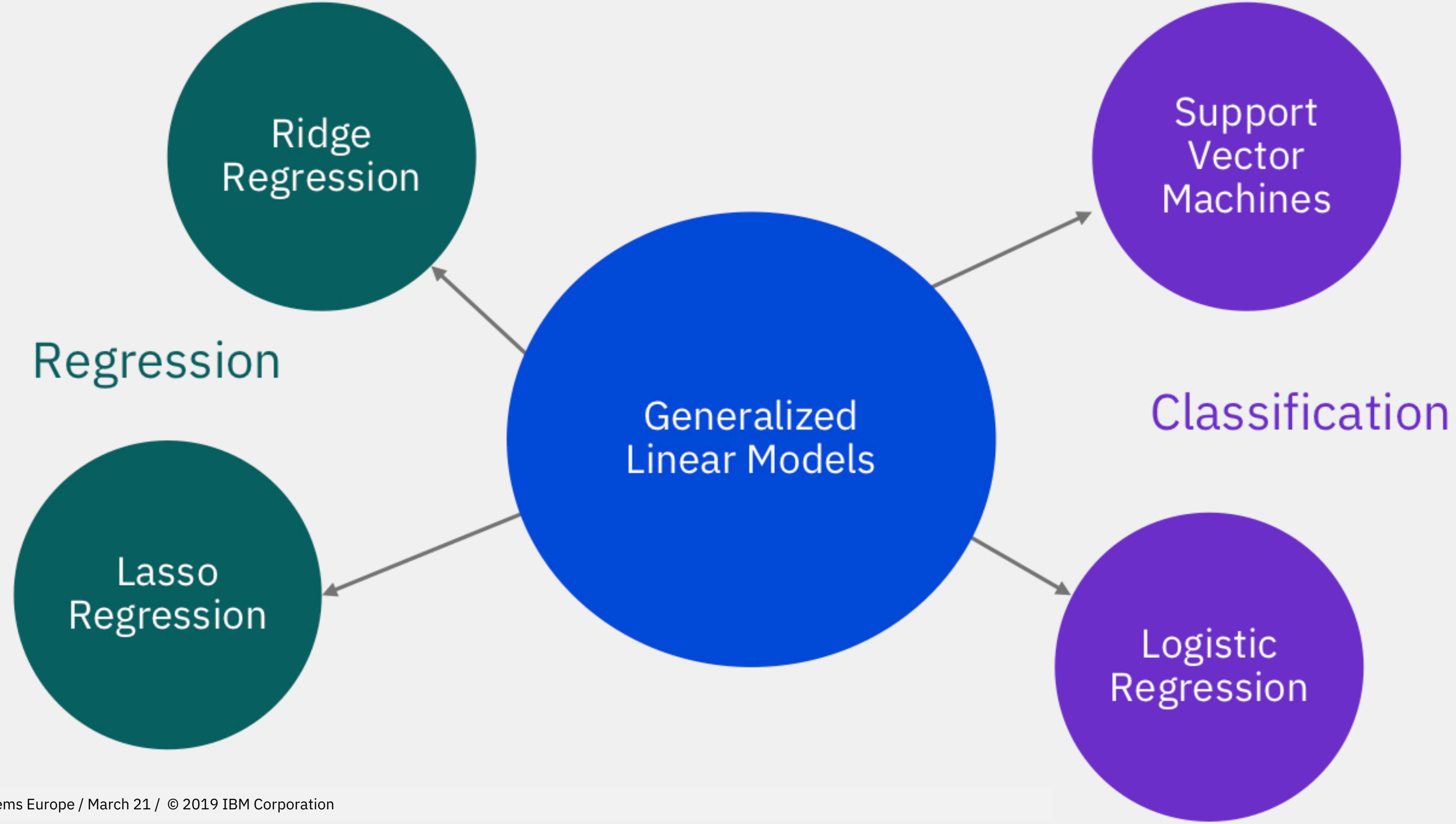
If training works by grabbing random data, modifications may not be necessary, although it should be verified that a different seed is being used for each process

Horovod with DDL

- Horovod is an open source distributed deep learning framework that works with TensorFlow and PyTorch
- DDL can now be used as a backend for Horovod, replacing MPI or NCCL
- Horovod is not shipped as part of PowerAI, a user must download Horovod in order to use it

PowerAI
Snap ML

What are GLMs?



Why are GLMs useful?

Fast Training

Can scale to datasets with billions of examples and/or features.

Less tuning

State-of-the-art algorithms for training linear models do not involve a step-size parameter.

Interpretability

New data protection regulations in Europe (GDPR) give E.U. citizens the right to “obtain an explanation of a decision reached” by an algorithm.

Linear models are naturally interpretable since they explicitly assign an importance to each input feature.

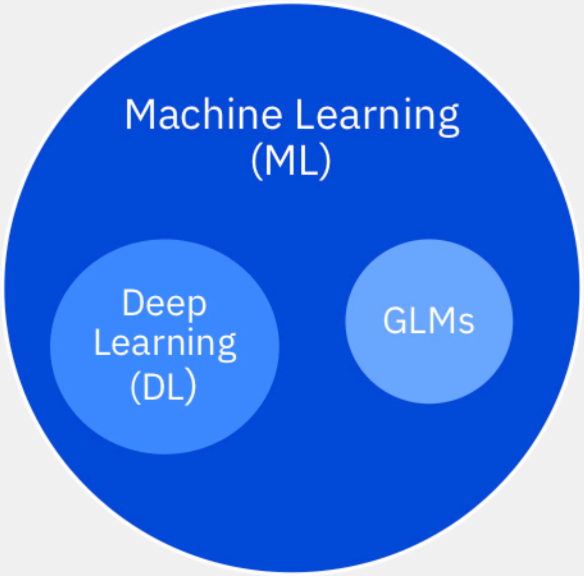
Widely used in industry

The Kaggle ”State of Data Science” survey asked 16,000 data scientists and ML practitioners what tools and algorithms they use on a daily basis.

37.6% of respondents use Neural Networks

63.5% of respondents use Logistic Regression

What is Snap Machine Learning?



Snap ML: A new framework for fast training of GLMs

Framework	Models	GPU Acceleration	Distributed Training	Sparse Data Support
scikit-learn	ML/{DL}	No	No	Yes
Apache Spark* MLlib	ML/{DL}	No	Yes	Yes
TensorFlow**	ML	Yes	Yes	Limited
Snap ML	GLMs	Yes	Yes	Yes

* The Apache Software Foundation (ASF) owns all Apache-related trademarks, service marks, and graphic logos on behalf of our Apache project communities, and the names of all Apache projects are trademarks of the ASF.
**TensorFlow, the TensorFlow logo and any related marks are trademarks of Google Inc.

Example: snap-ml-local

```python
# Load data
from sklearn.datasets import load_from_svmlight_format
X, y_ = load_from_svmlight_format(filename_train)

# Train/test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)

# Create the logistic regression
if(use_snap_ml):
    from snap_ml import LogisticRegression
    lr = LogisticRegression(device_ids=[0,1])
else:
    from sklearn.linear_model import LogisticRegression
    lr = LogisticRegression()

# Training
lr.fit(X_train, y_train)

# Inference
proba_test = lr.predict_proba(X_test)

# Evaluate logarithmic loss on test set
from sklearn.metrics import log_loss
test_loss = log_loss(y_test, proba_test)
```

Acceleration existing scikit-learn applications by changing only 2 lines of code.

Example: snap-ml-mpi

Describe application using high-level Python code.

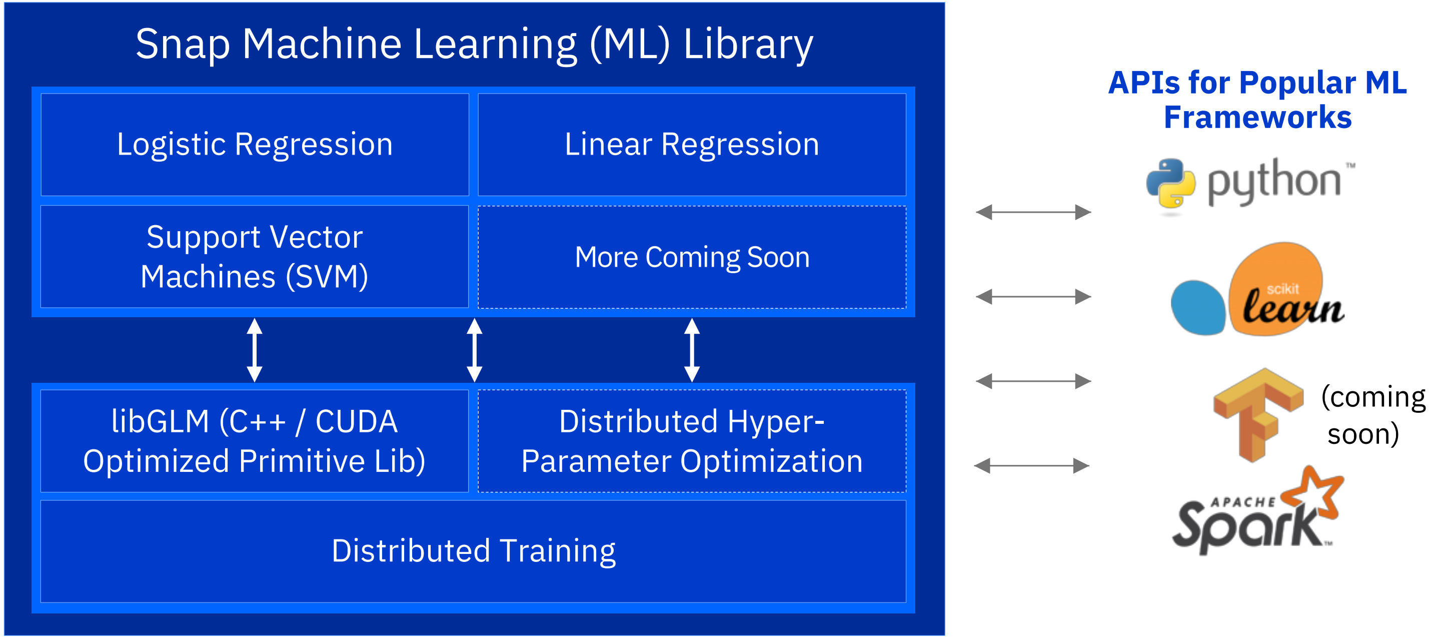
```
# Load data
from snap_ml_mpi.Loaders import load_from_snap_format
train_data = load_from_libsvm_format(train_filename)
test_data  = load_from_libsvm_format(test_filename)

# Create the logistic regression
from snap_ml_mpi import LogisticRegression
lr = LogisticRegression(max_iter=200, dual=True, device_ids=[0,1,2,3], num_threads=128)

# Training
lr.fit(train_data)

# Inference
proba_test = lr.predict_proba(test_data)

# Evaluate logarithmic loss on test set
from snap_ml_mpi.Metrics import log_loss
test_loss = log_loss(y_test, proba_test)
```

Launch application on 4 nodes using mpirun (4 GPUs per node):

```
$ mpirun -n 4 -rf myrankfile python my_app.py
```

IBM Snap ML part of PowerAI Base

Distributed GPU-Accelerated Machine Learning Library



Snap.ML updates in PowerAI 1.6.1

Accelerated Machine Learning with SnapML

... Backend integration with scikit-learn (pai4sk) with Spark distribution

... Backend integration with scikit-learn with MPI

... Support for cuDF (GPU dataframes) as input to snapML APIs

... Nvidia RAPIDs integration with cuML

Snap ML: Training Time Goes From An Hour to Minutes

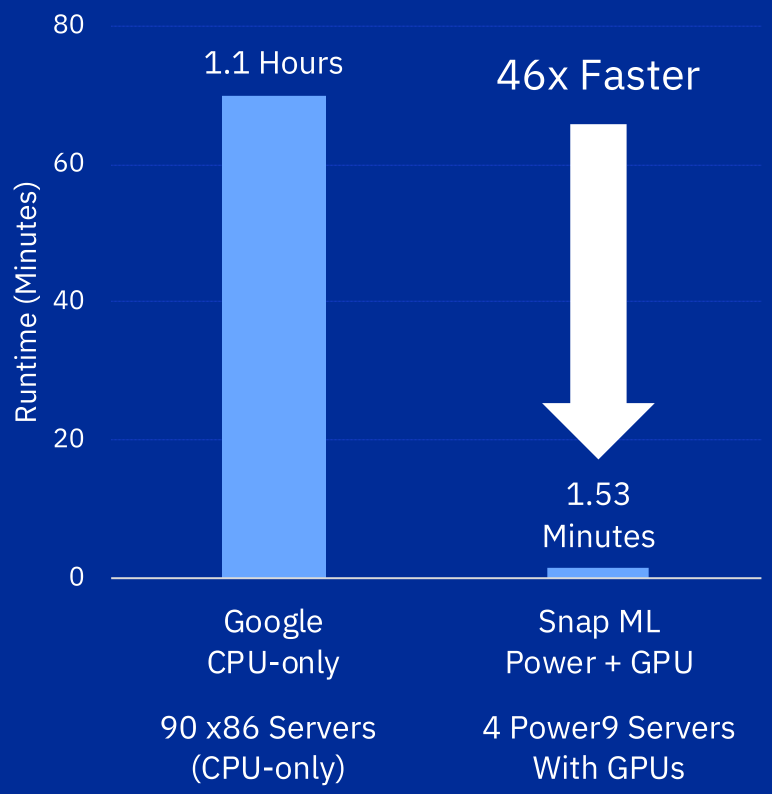
46x faster than previous record set by Google

Workload: Click-through rate prediction for advertising

Logistic Regression Classifier in Snap ML using GPUs vs TensorFlow using CPU-only

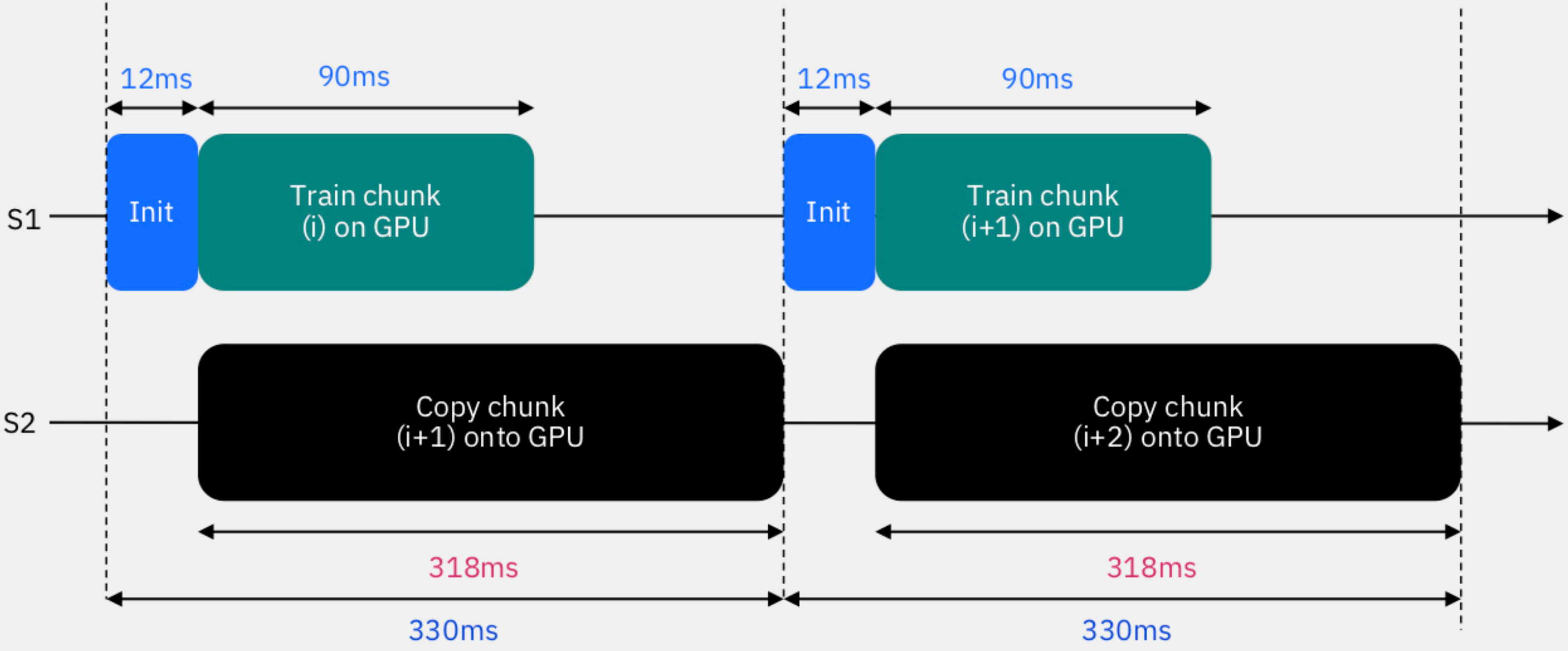
Dataset: Criteo Terabyte Click Logs (http://labs.criteo.com/2013/12/download-terabyte-click-logs/)
4 billion training examples, 1 million features
Model: Logistic Regression: TensorFlow vs Snap ML
Test LogLoss: 0.1293 (Google using Tensorflow), 0.1292 (Snap ML)
Platform: 89 CPU-only machines in Google using Tensorflow versus 4 AC922 servers (each 2 Power9 CPUs + 4 V100 GPUs) for Snap ML
Google data from [this Google blog]

Logistic Regression in Snap ML (with GPUs) vs TensorFlow (CPU-only)



Out-of-core performance (PCIe Gen3)

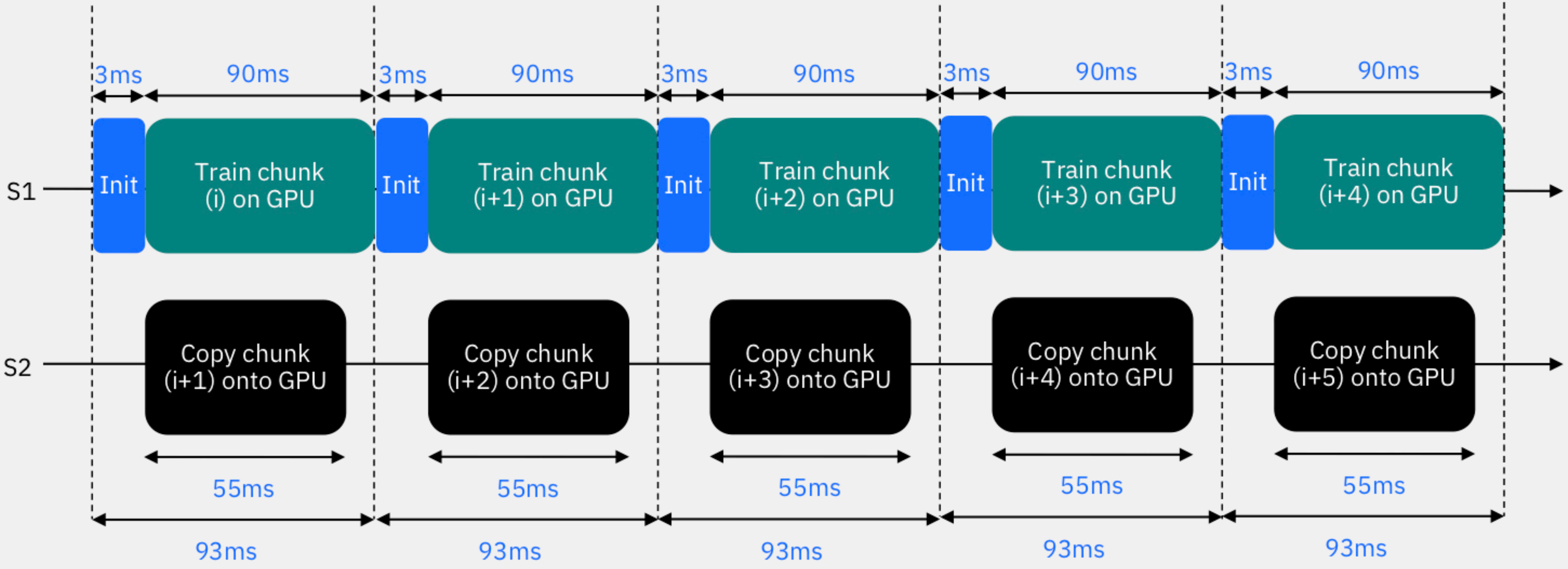
Dataset	Examples	Nodes	Total GPUs	Network	CPU-GPU interface
criteo-tb	200 million	1x Intel Xeon* Gold 6150	1x V100	n/a	PCI Gen3



*Intel Xeon is a trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Out-of-core performance (NVLINK 2.0)

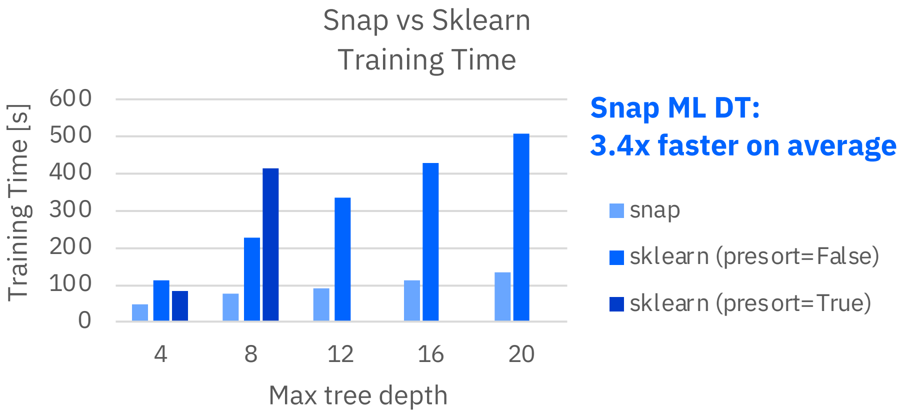
Dataset	Examples	Nodes	Total GPUs	Network	CPU-GPU interface
criteo-tb	200 million	1x Power AC922 server	1x V100	n/a	NVLINK 2.0

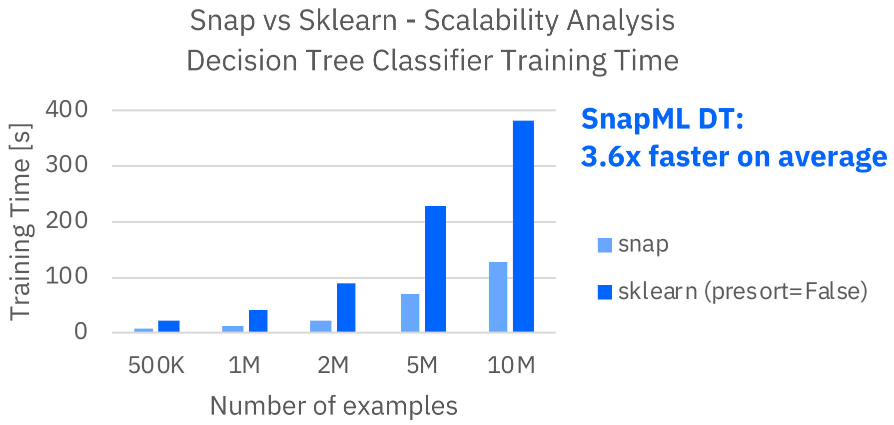


Decision Tree Classifier

Decision Tree is a new experimental feature of Snap ML (currently on CPU only)

Snap ML vs. scikit-learn Decision Tree (single AC922 node, CPU-only)





Model Accuracy

max_depth	snap	sklearn (presort=False)	sklearn (presort=True)
4	0.6455	0.6455	0.6455
8	0.6864	0.6864	0.6864
12	0.7034	0.7032	n/a
16	0.7079	0.707	n/a
20	0.6972	0.6969	n/a

Model Accuracy

no. examples	snap	sklearn (presort=False)	sklearn (presort=True)
500K	0.685	0.6851	
1M	0.6857	0.6858	
2M	0.6859	0.6859	
5M	0.6864	0.6864	
10M	0.686	0.6782	

Watson ML Accelerator
Hyperparameter Tuning

Real time monitoring of hyper parameters in PowerAI Enterprise

ost Cluster

Search Search

Jobs

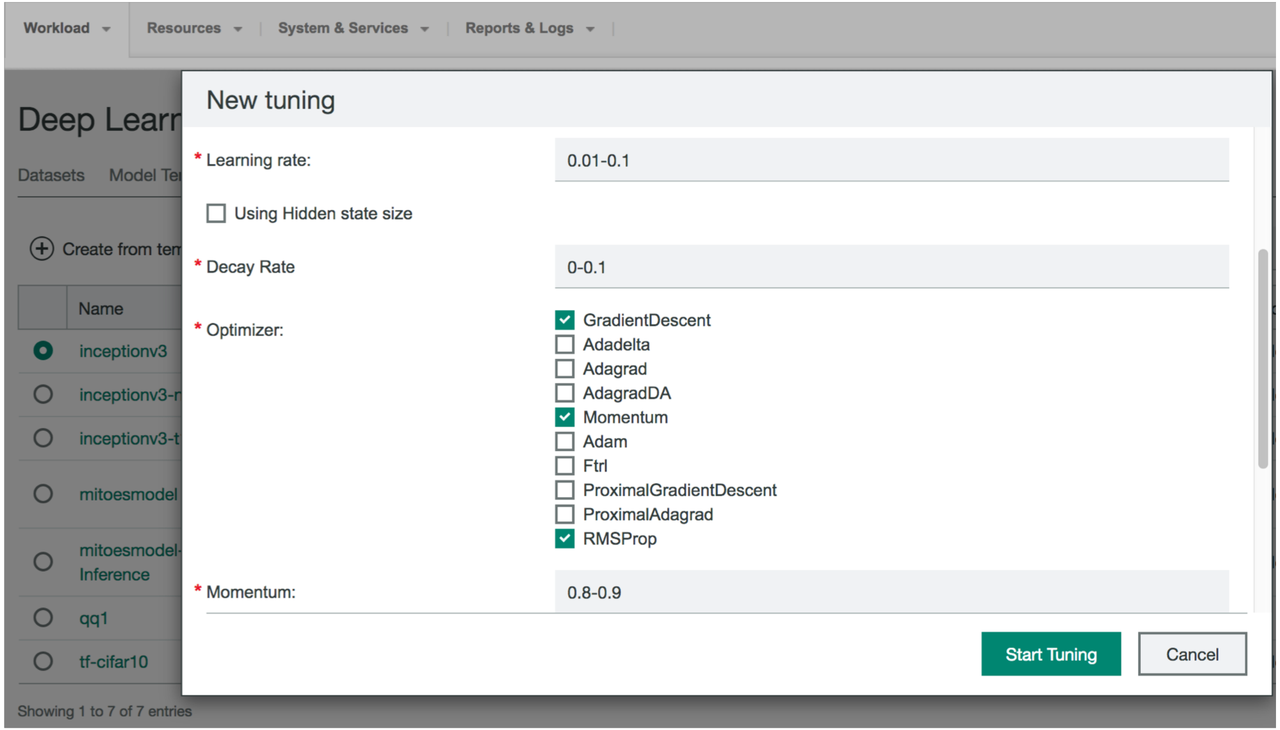
Start: End: Time Filter

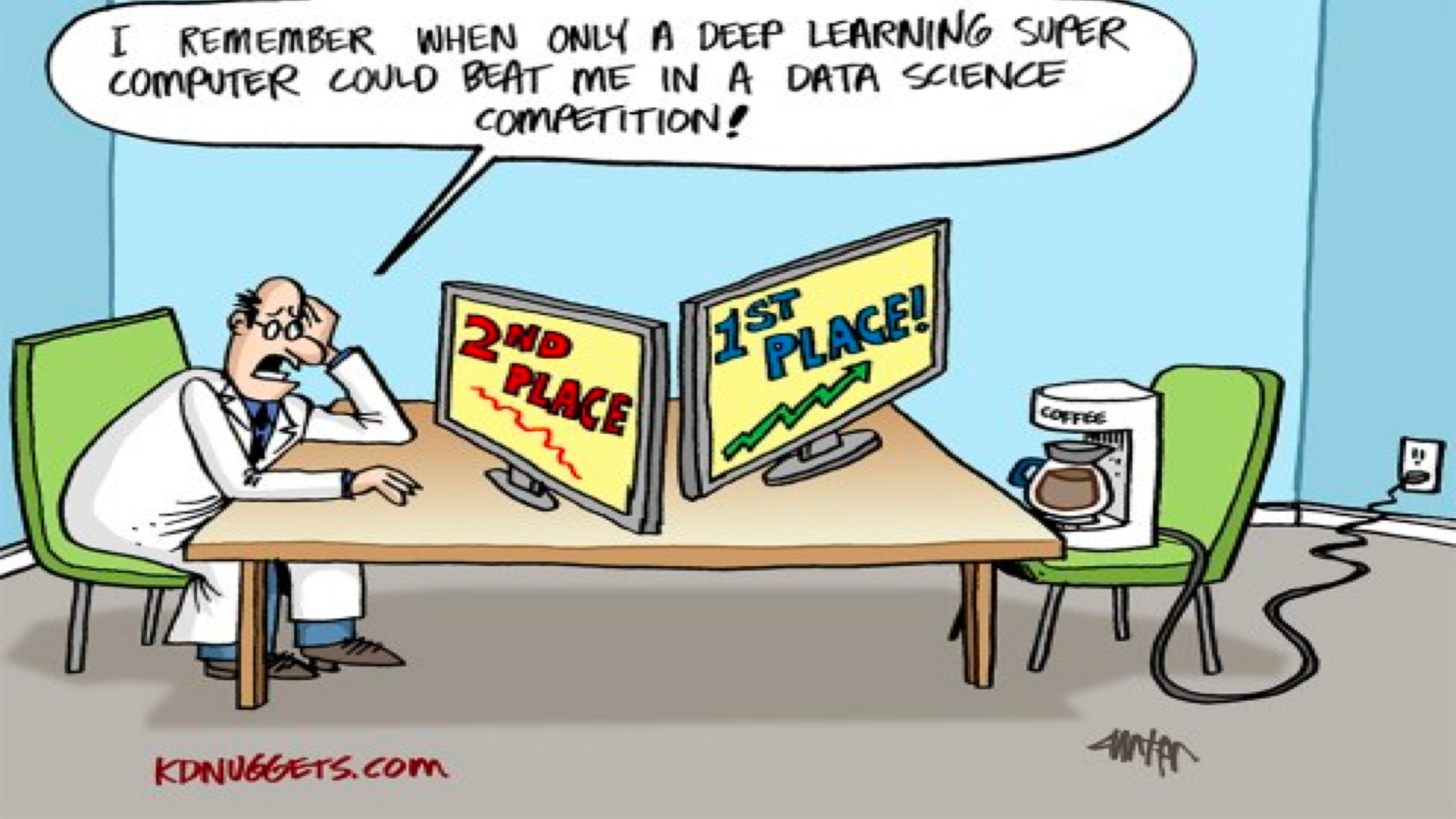
s found

lication	Cluster's master	Type	GPU uses	Status
pp-20170308061821-0242	172.17.0.3	CaffeOnSpark	1	0.061788617886178863
app-20170308061443-0241	172.17.0.3	CaffeOnSpark	1	1
app-20170308061153-0240	172.17.0.3	CaffeOnSpark	1	1
app-20170308061016-0239	172.17.0.3	CaffeOnSpark	0	1
app-20170308060849-0238	172.17.0.3	CaffeOnSpark	0	1
app-20170308060058-0237	172.17.0.3	CaffeOnSpark	1	1

Hyper-parameter Tuning/Search in PowerAI Enterprise

Hyper-parameters
- Learning rate
- Decay rate
- Batch size
- Optimizer:
  - GradientDecedent,
  - Adadelta,
  - Momentum,
  - RMSProp
  - .....
- Momentum (for some optimizers)
- LSTM hidden unit size (for models which use LSTM)




I REMEMBER WHEN ONLY A DEEP LEARNING SUPER COMPUTER COULD BEAT ME IN A DATA SCIENCE COMPETITION!
2ND PLACE
1ST PLACE!
COFFEE
KDNUGGETS.COM

Who are the typical Personas for computer vision solutions ?

Iris
Image Analyst

- Knows datasets
- Searches for tools that automate data labeling
- Creates taxonomy, data hierarchy
- Finds insights from images & videos
- Provides curated datasets to data scientist
- Collaborates on data curation & labeling

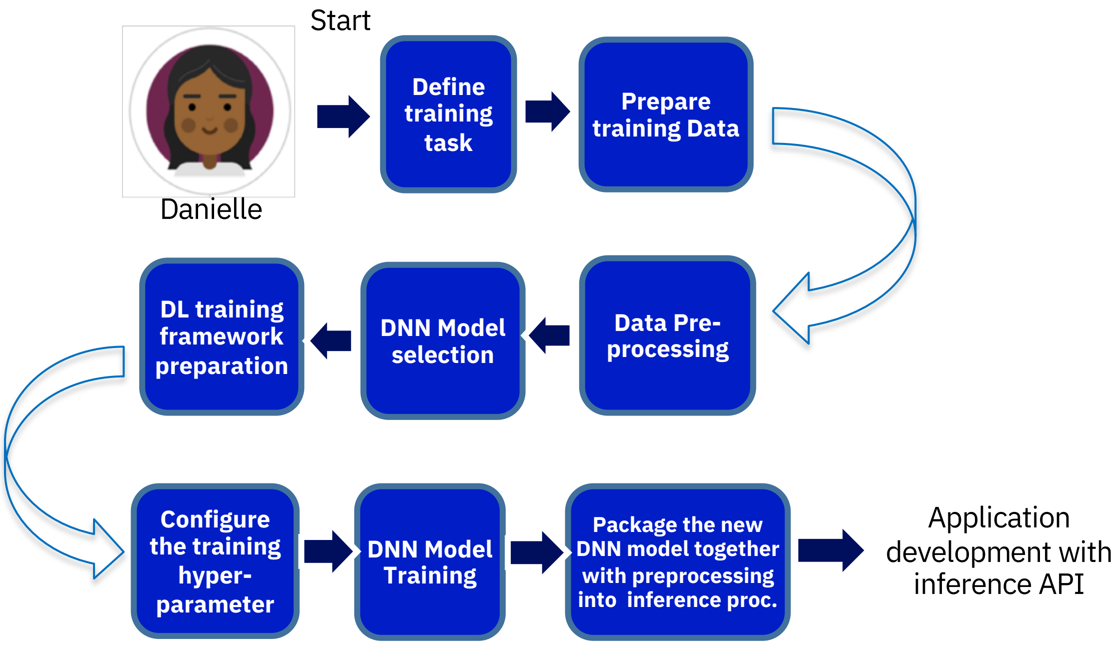
Danny
Developer

- Iterates constantly
- Writes code, but not familiar with TensorFlow
- Has taken 1 day ML/DL online course
- Delivers apps quickly
- Works with business stakeholders

Danielle
Data Scientist

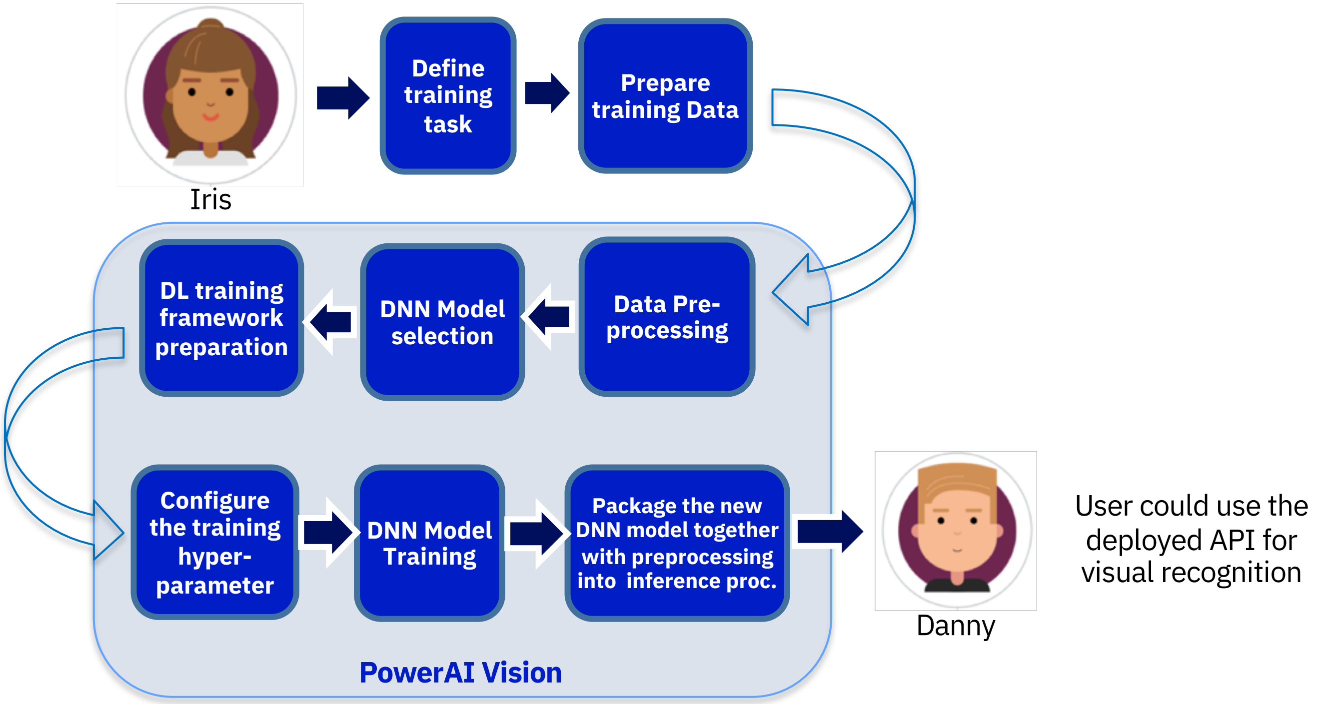
- Uses datasets to train models
- Develops customer models to classify and detect objects
- Creates models
- Always experimenting and fiddling
- Finds insights in image data

Steps for Deep Learning Development



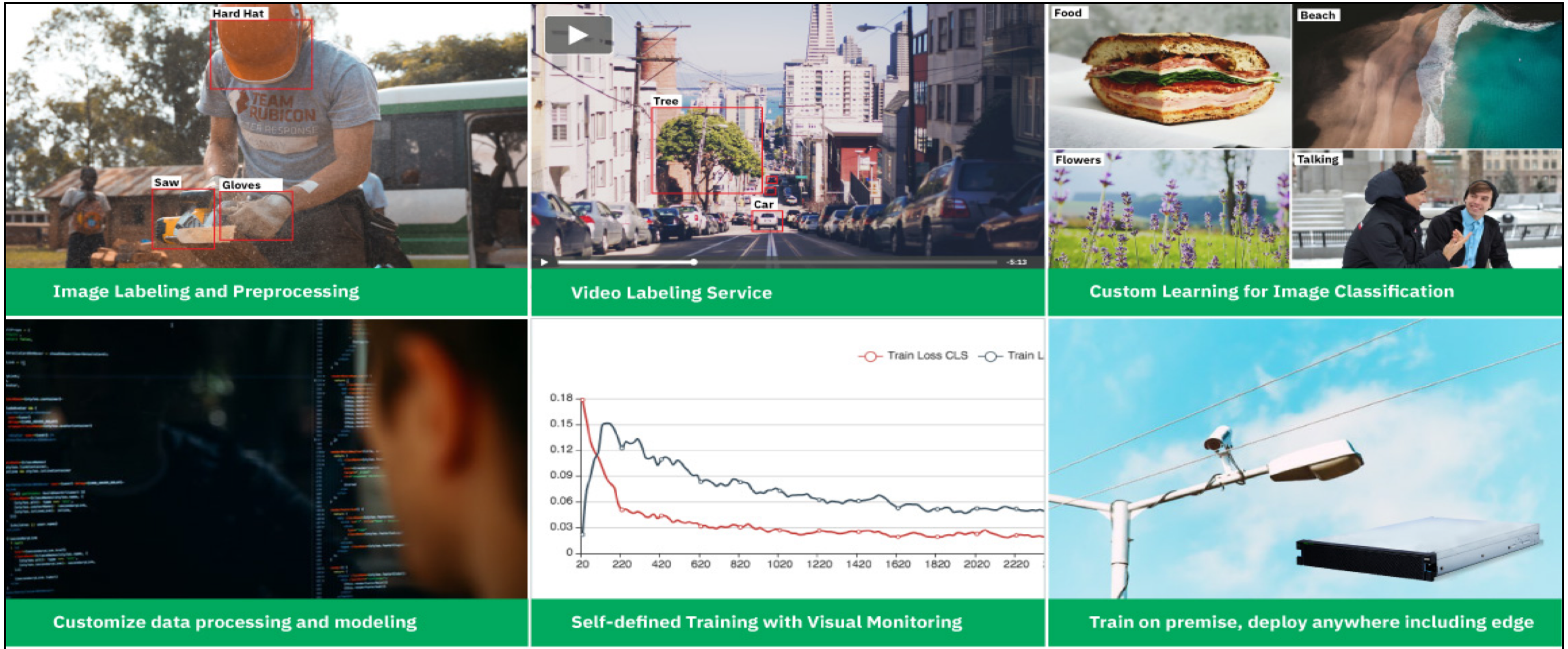
IBM PowerAI Vision

Simplify Deep Learning Adoption



IBM PowerAI Vision

Lowers the barriers for creating Computer Vision related AI solutions.

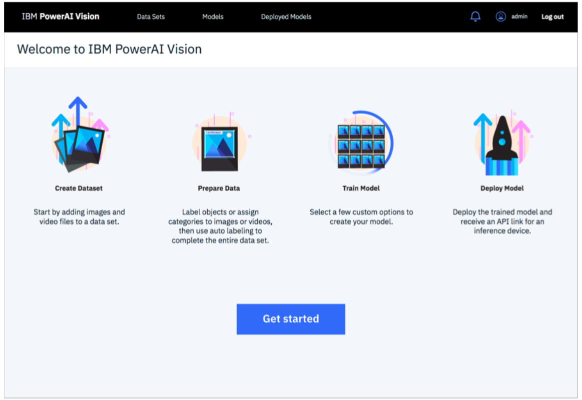


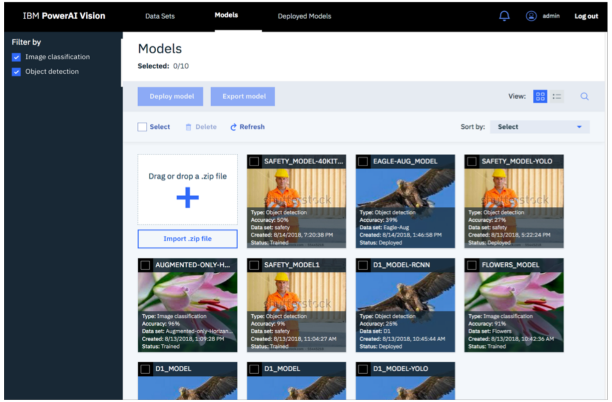
Management Services

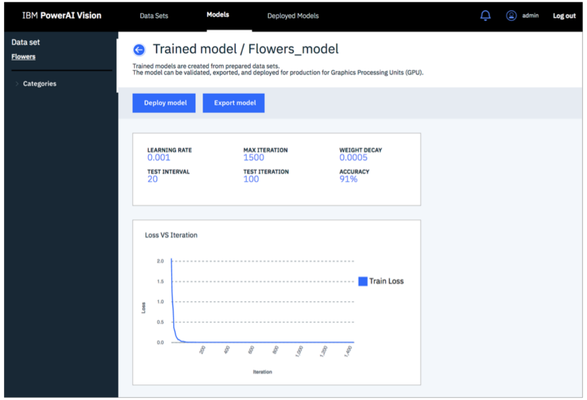
- Image Processing Management
- Data Label Management
- Dataset Management
- Training Task Management
- Model Management
- Inference API Management

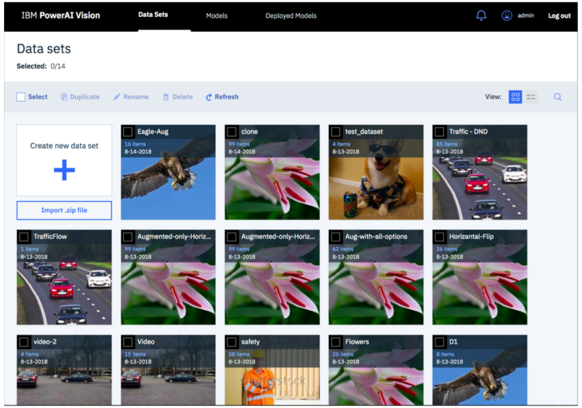
IBM PowerAI Vision

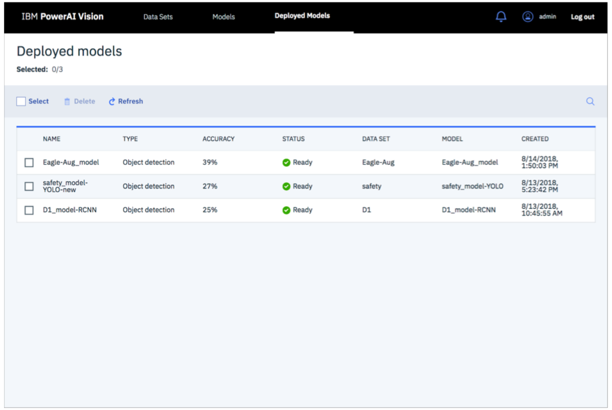
Lowers the barriers for creating Computer Vision related AI solutions.

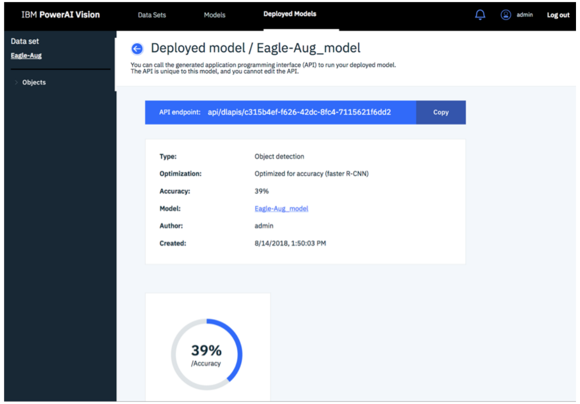






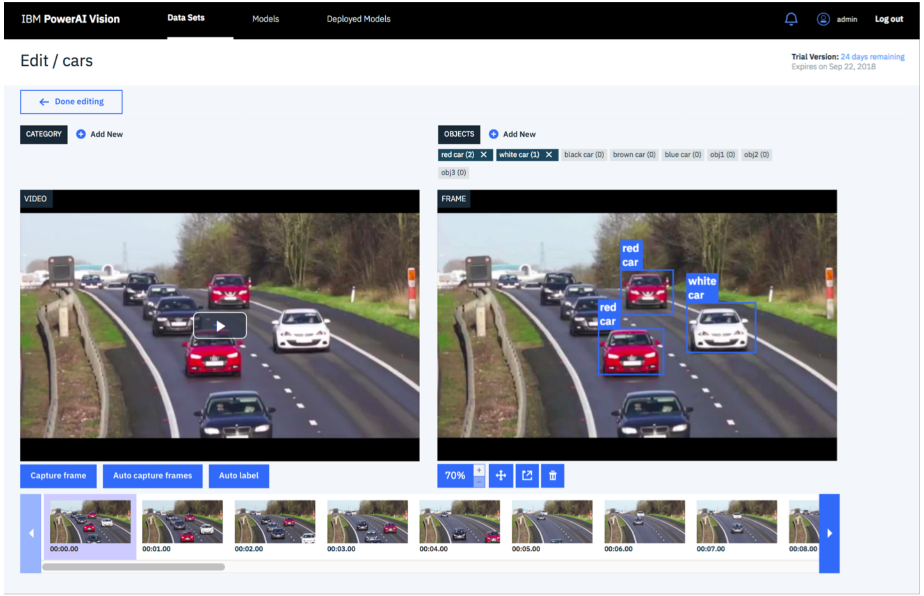


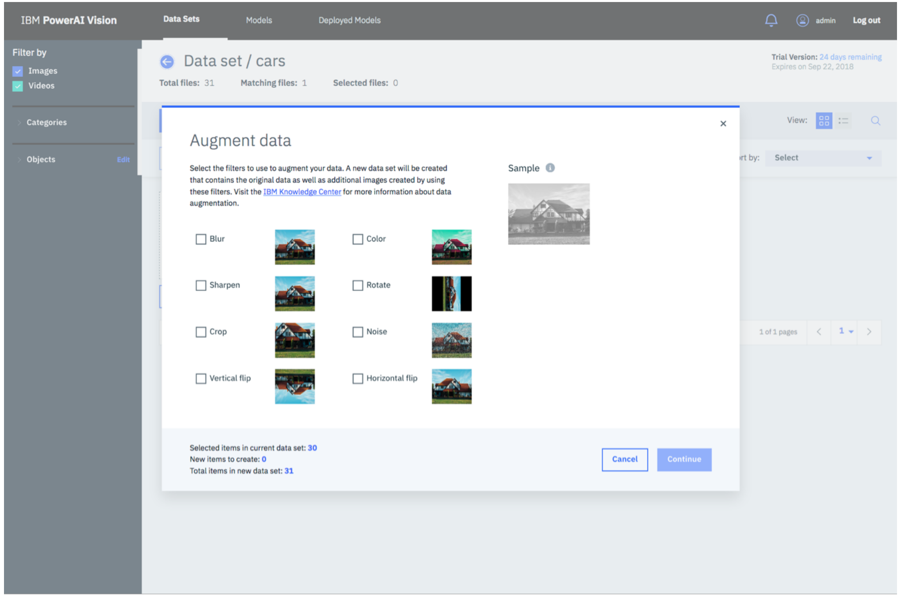




IBM PowerAI Vision

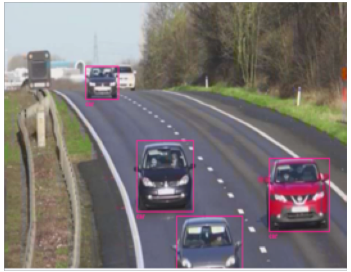
Lowers the barriers for creating Computer Vision related AI solutions.





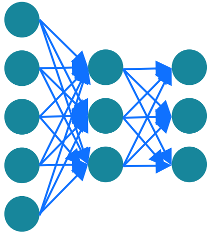
Semi-Automatic Labeling from video content

Manually Label



Define Labels
Manually Label Some Images / Video Frames

Train DL Model



Use Trained DL Model

Run Trained DL Model on Entire Input Data to Generate Labels

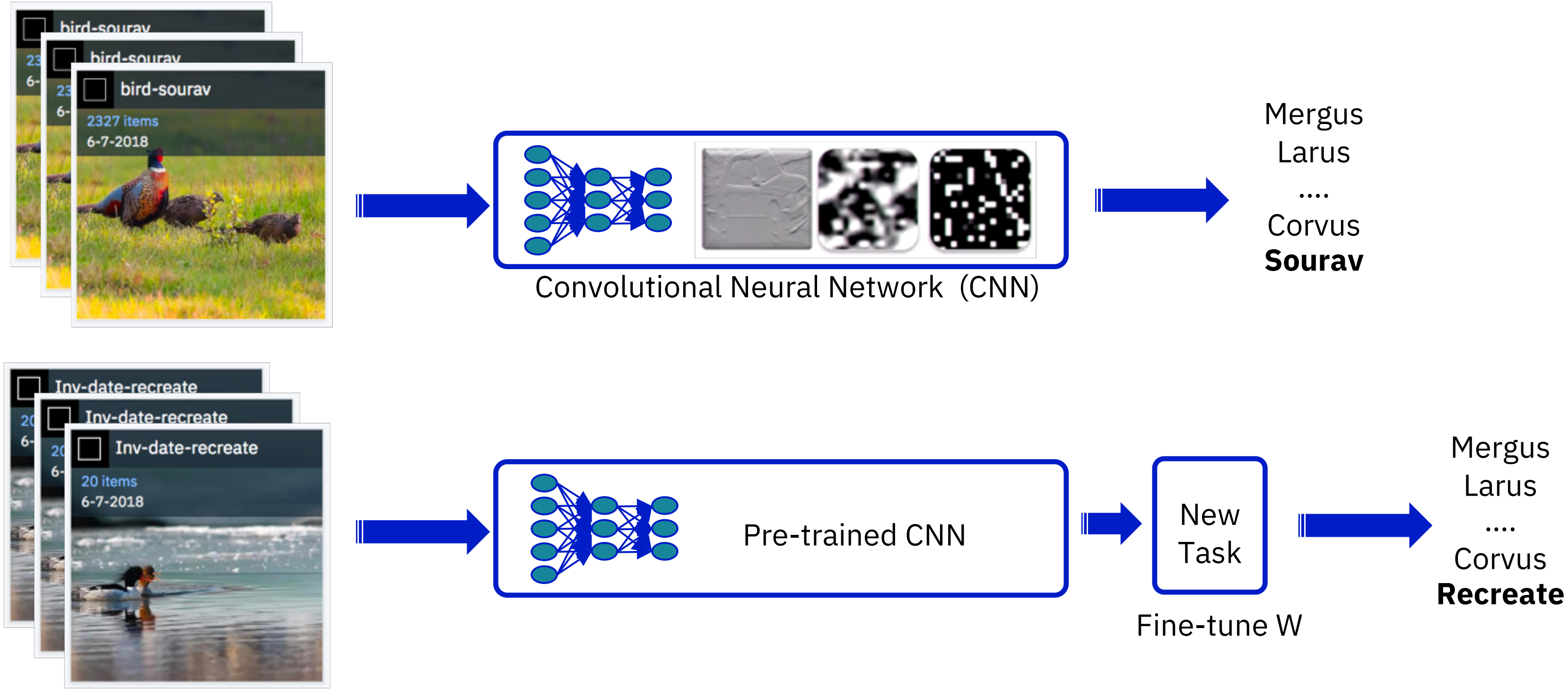
Correct Labels on Some Data

Manually Correct Labels on Some Data

Repeat Till Labels Achieve Desired Accuracy

Delivered Pre-Trained Models

Time and Data Matters



IBM PowerAI Vision: Deep Learning Development Platform for Computer Vision

Vision Recognition Service Layer

Image Labeling and Preprocessing

Video Labeling Service

Custom Learning for Image Classification

Custom Learning for Object Detection

Inference API deployment

Service Management Layer

- Image preprocessing management
- Data label management
- Data set management
- Training task management
- Model management
- Inference API management

ML/DL Computation Layer

- Data preprocessing component
- Model Training Component (Caffe, and others)
- Inference Component (Caffe, and others)

Resource management layer (CPU/GPU/FPGA) (Docker, Kubernetes)

- Docker (Open Power)
- Accelerator (GPU/FPGA)
- Data Store (distributed FS and object store)
- Network

PowerAI Vision APIs

Inference APIs for Object Detection (example)

Developer could use these APIs for object detection with the deployed model in PowerAI Vision from any IP device

http://IP:PORT/ (of the deployed inference instance)

/test

GET: Only to test if the monitor service is running.

/detect_url

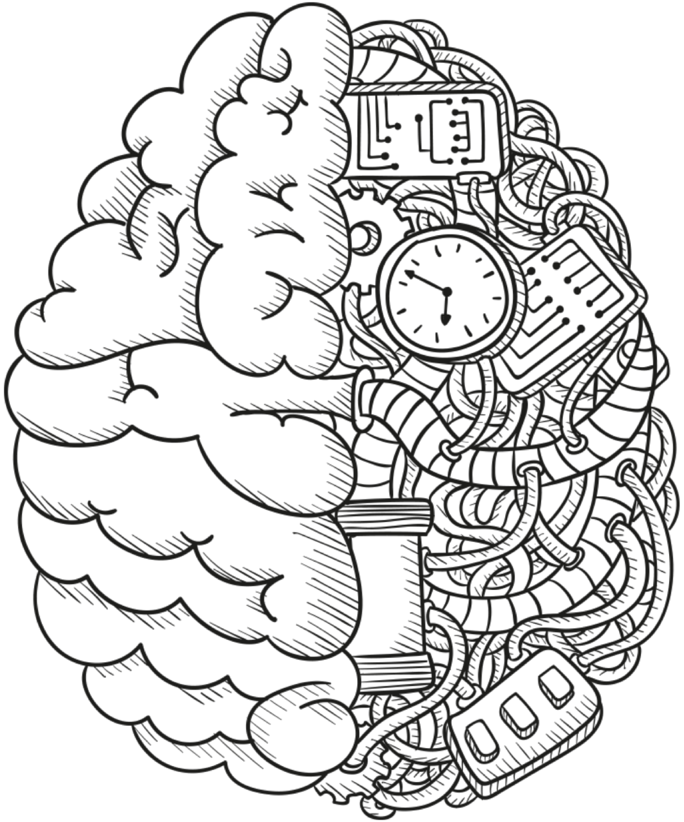
GET: Upload image with image url and detect objects

/detect_upload

POST: Post image file and do the object detection

Inference return:

{'confidence': 0.9038739204406738, 'ymax': 145, 'label': 'badge', 'xmax': 172, 'xmin': 157, 'ymin': 123}



Thank you

Ing.Florin Manaila
Cognitive Systems Europe
—
florin.manaila@de.ibm.com

ibm.com

Notices and disclaimers

U.S. Government Users Restricted Rights — use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed "as is" without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply."

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimers continued

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

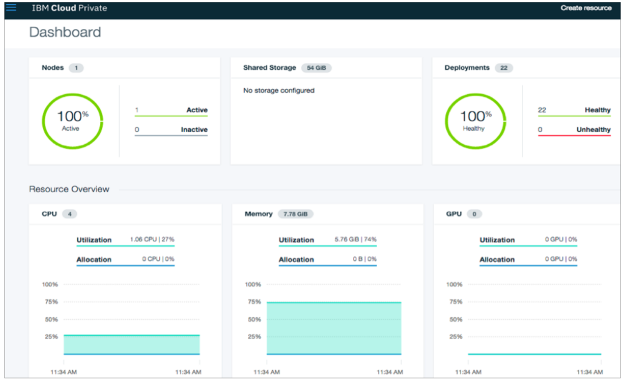
IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.

Enterprise AI your way

Deep Learning Containers on AC922 with Kubernetes

SCENARIO

ON-PREM or SaaS



IBM Cloud Private

Management

ON-PREM

IBM Data Science Experience

H2O.ai

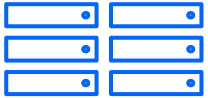
+ custom GPU accelerated containers

IBM PowerAI Base

IBM PowerAI Enterprise

IBM PowerAI Vision

IBM Cloud Private Workers

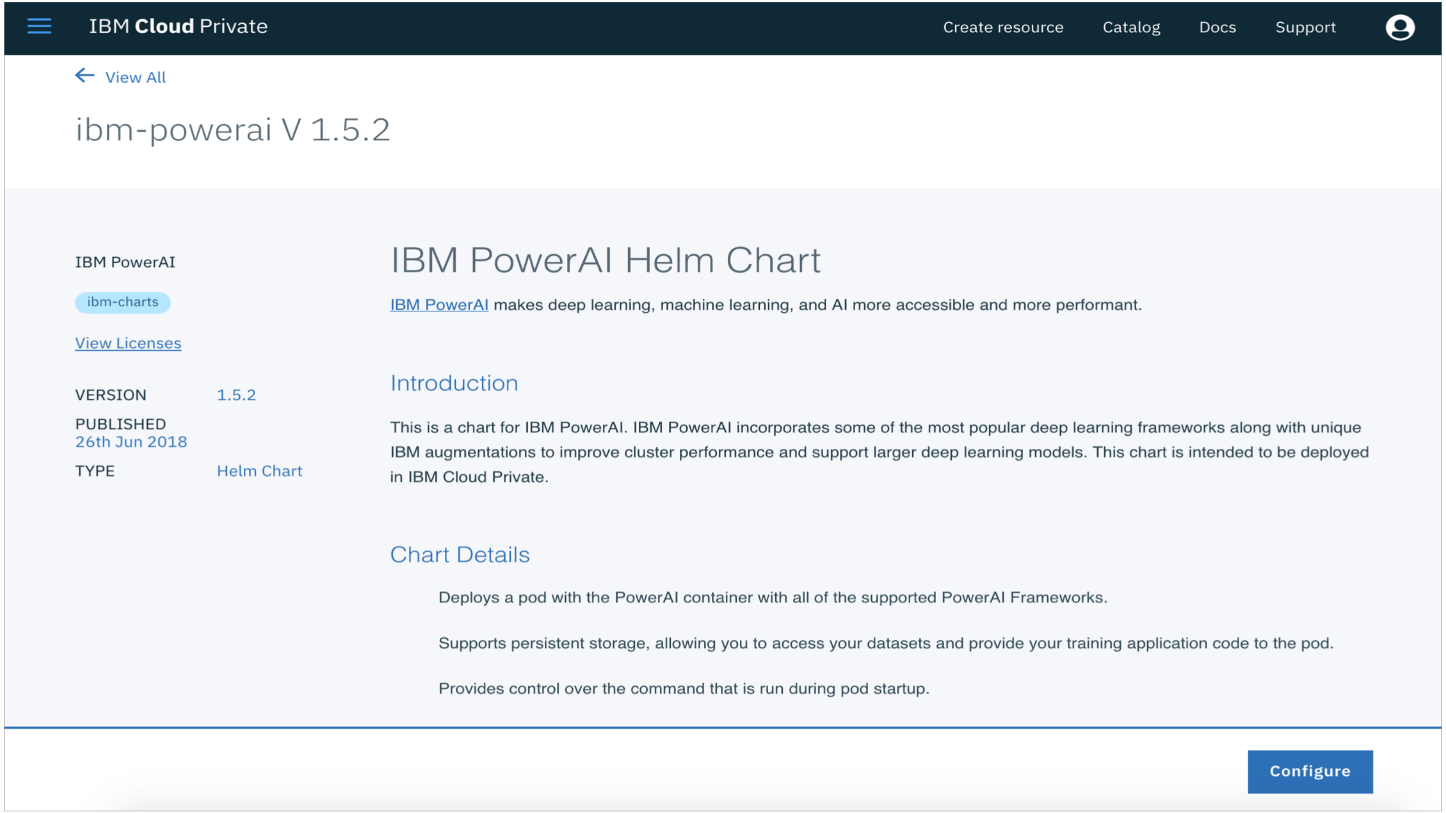


IBM AC922 Servers

IBM Storage

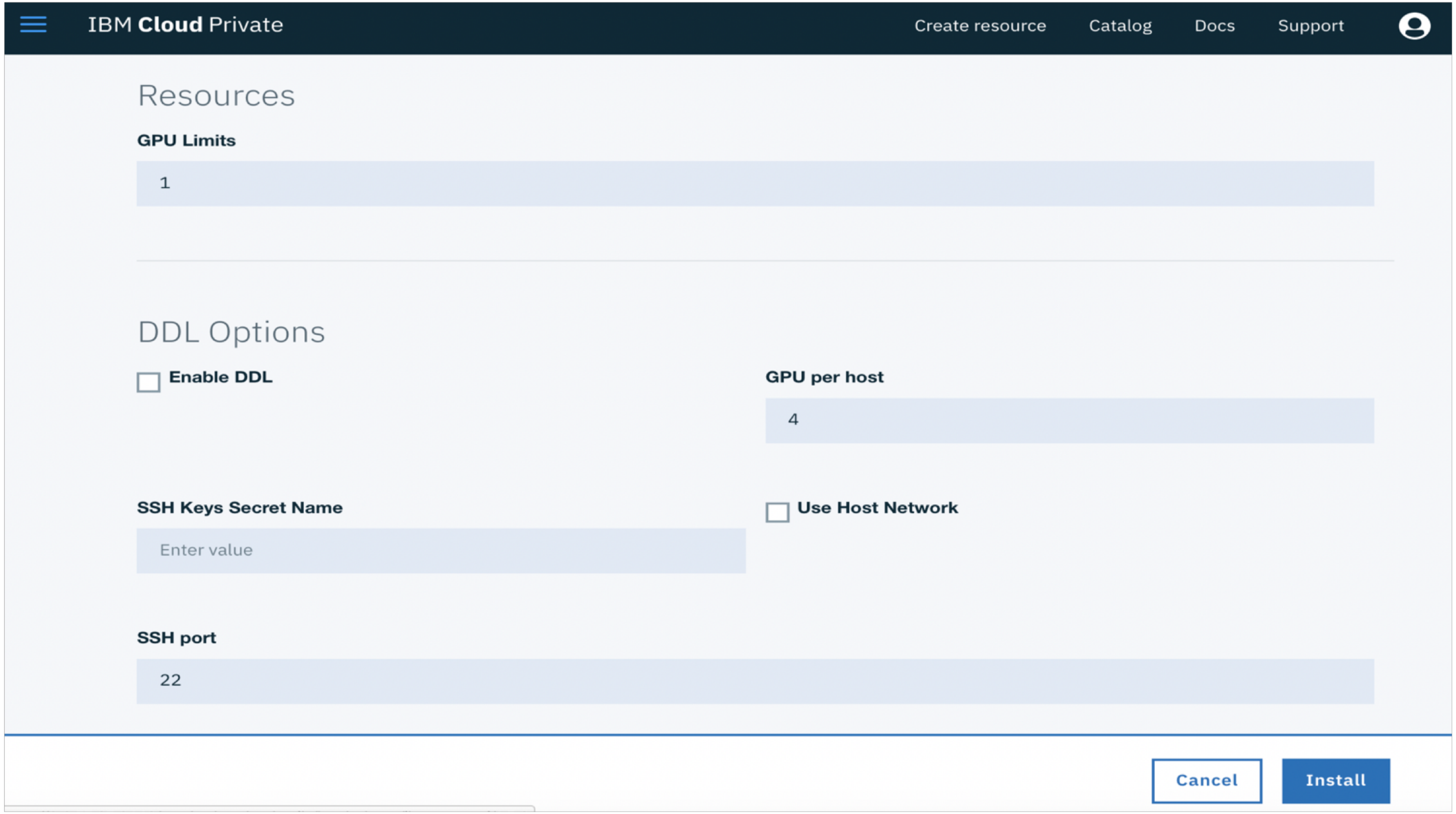
PowerAI on IBM Cloud Private

Deployed on AC922



PowerAI on IBM Cloud Private

Deployed on AC922



H2O Driverless AI on IBM Cloud Private

Deployed on AC922