



Accelerate Your Machine Learning Applications on the CMC FPGA/GPU Cluster

July 31st, 2019

Note: participants are muted upon entering the webinar.
Please use the chat feature to ask questions.

Dr. Yassine Hariri

Hariri@cmc.ca

Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- Use Case : CNN architecture and training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

Agenda



- **CMC Microsystems**
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- Use Case : CNN architecture and training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

What is CMC and what is its role?



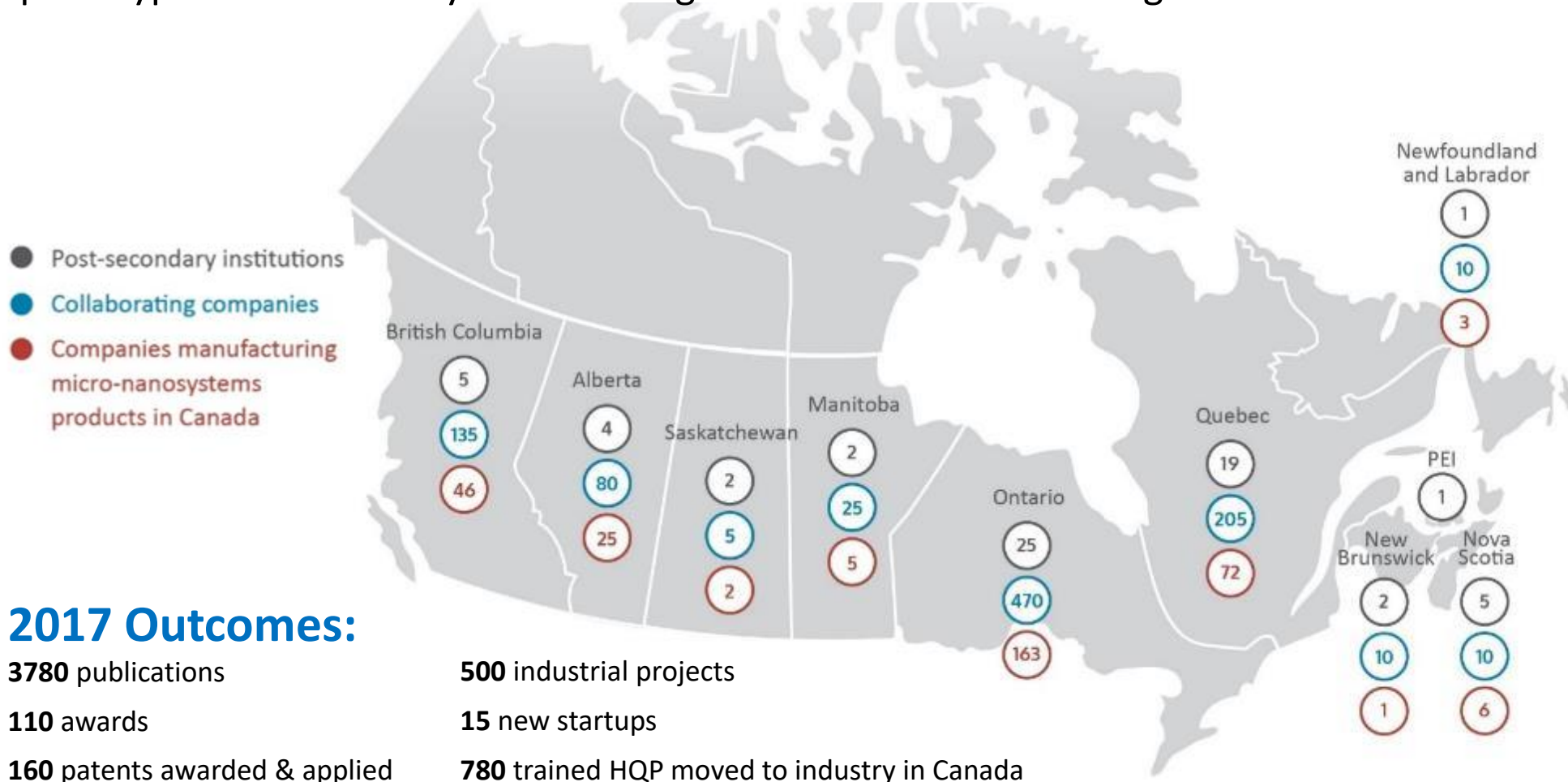
- Not for profit – federally incorporated 1984
- Manages Canada's National Design Network[®]
- Delivers micro-nano innovation capabilities across Canada



Canada's National Design Network®



A Canada-wide collaboration between **66** universities/colleges to connect **10,000** academic participants with **950** companies to design, make and test micro-nanosystem prototypes. CMC Microsystems manages Canada's National Design Network®.



Annually:

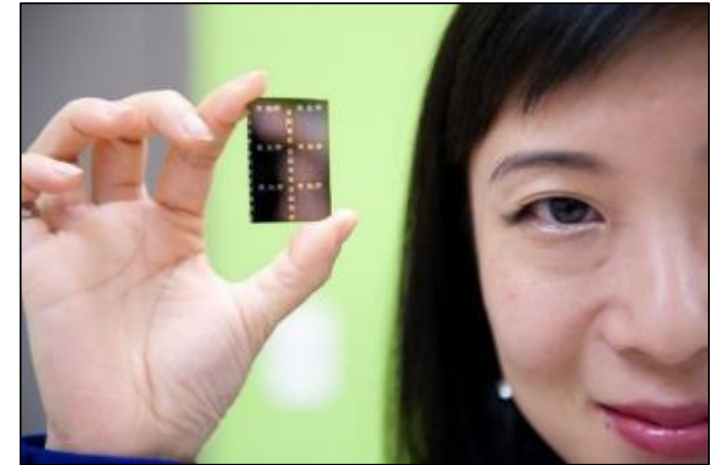
- 1200** connected professors
- 4200** researchers on professors' teams
- 5700** users of computer aided design tools
- 300** physical prototypes
- 80** test equipment loan items otherwise unaffordable to users

Lowering Barriers to Technology Adoption



CMC delivers key services to increase researchers' and companies' innovation capability in Canada:

- Design tools (software)
- Fabrication services to create working prototypes
- Equipment and services for prototype testing
- Platform technologies
- Training, support, networking
- Technology plan and roadmap



LOWERING BARRIERS TO TECHNOLOGY ADOPTION



CAD



Services for making working prototypes

- ✓ Selection of high-performance Computer Aided Design (CAD) tools and design environments
- ✓ Available via desktop or through CMC Cloud
- ✓ User guides, application notes, training materials and courses

 [CMC.ca/CAD](https://cmc.ca/CAD)

FAB



Services for making working prototypes

- ✓ Multi-project wafer services with affordable access to foundries worldwide
- ✓ Fabrication and travel assistance to prototype at a university-based lab
- ✓ Value-added packaging and assembly services
- ✓ In-house expertise for first-time-right prototypes

 [CMC.ca/FAB](https://cmc.ca/FAB)

LAB



Device validation to system demonstration

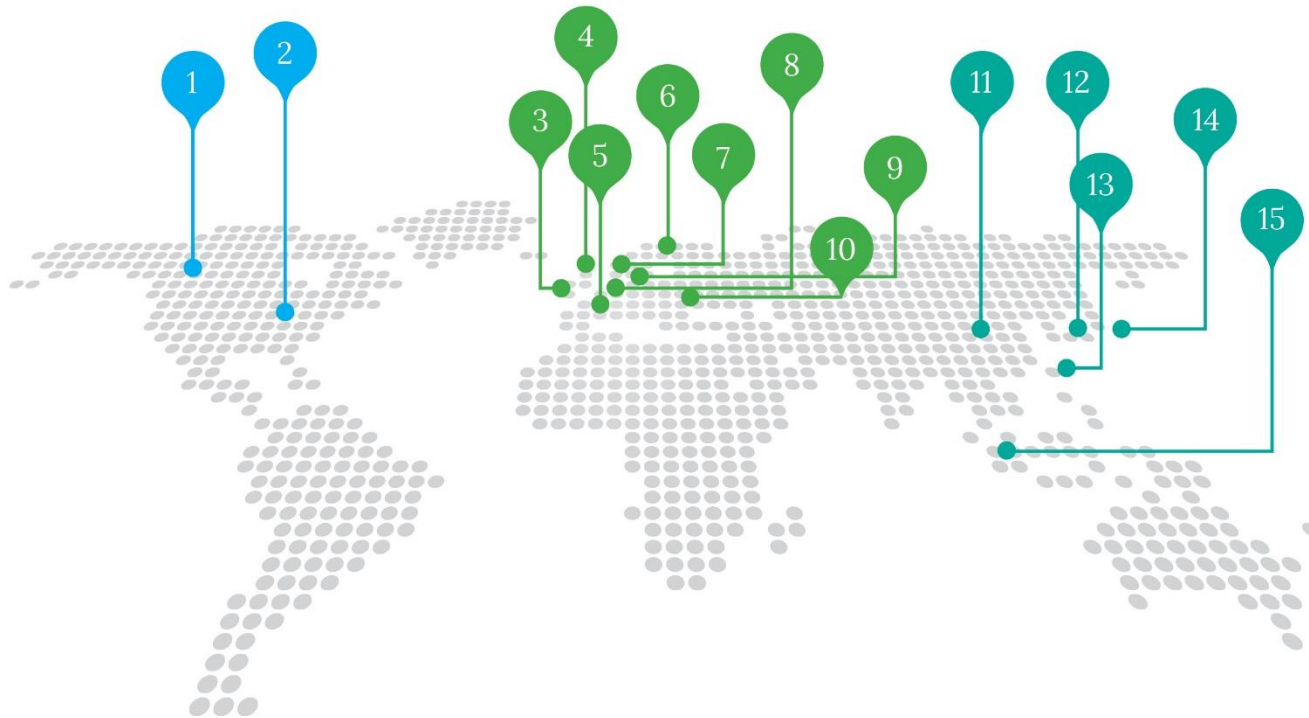
- ✓ Access to platform-based microsystems design and prototyping environments
- ✓ Access to test equipment on loan
- ✓ Access to contract engineering services

 [CMC.ca/LAB](https://cmc.ca/LAB)

ENGAGING STRATEGICALLY in Canada and worldwide



Strategic Engagements, Global Partners



North America

- | | |
|-----------|---|
| 1. Canada | 14 CAD 8 FAB 13 LABs 19 Systems & Components 42 University MNT LABs |
| 2. USA | 1 Co-operative Initiative 15 CAD 5 FAB 11 LABs 8 Systems & Components |

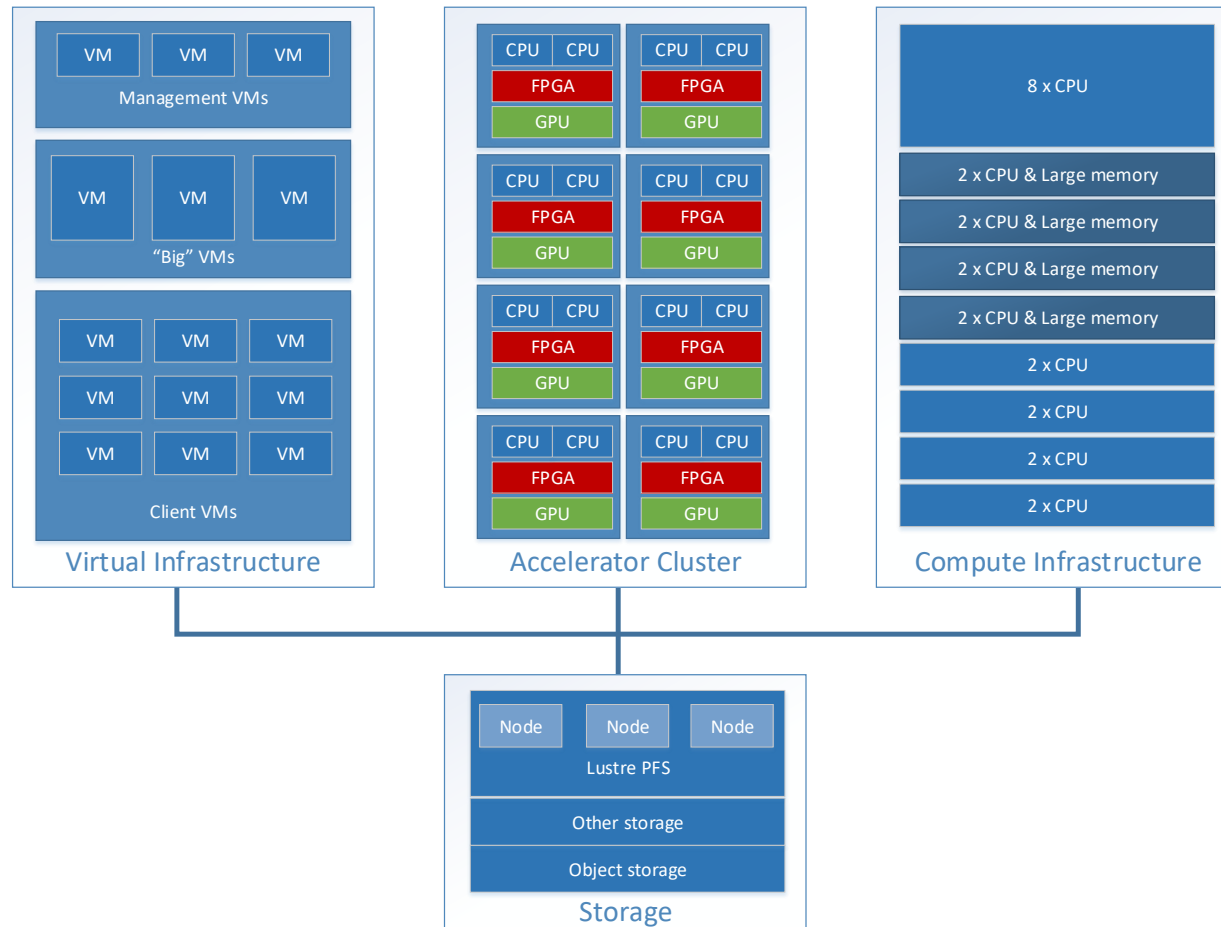
Asia

- | | |
|-----------------|--|
| 11. China | 1 Co-operative Initiative |
| 12. South Korea | 1 Co-operative Initiative |
| 13. Taiwan | 1 Co-operative Initiative 2 FAB 1 LAB 2 Systems & Components |
| 14. Japan | 1 Co-operative Initiative |
| 15. Singapore | 3 FAB |

Europe 1 Co-operative Initiative

- | | |
|----------------|-----------------------------------|
| 3. Ireland | 1 FAB |
| 4. UK | 1 CAD 1 Systems & Components |
| 5. France | 2 FAB 1 Co-operative Initiative |
| 6. Sweden | 1 CAD |
| 7. Netherlands | 2 FAB |
| 8. Belgium | 1 FAB |
| 9. Germany | 1 CAD 2 FAB |
| 10. Austria | 1 FAB |

CMC Cloud: Unified Architecture



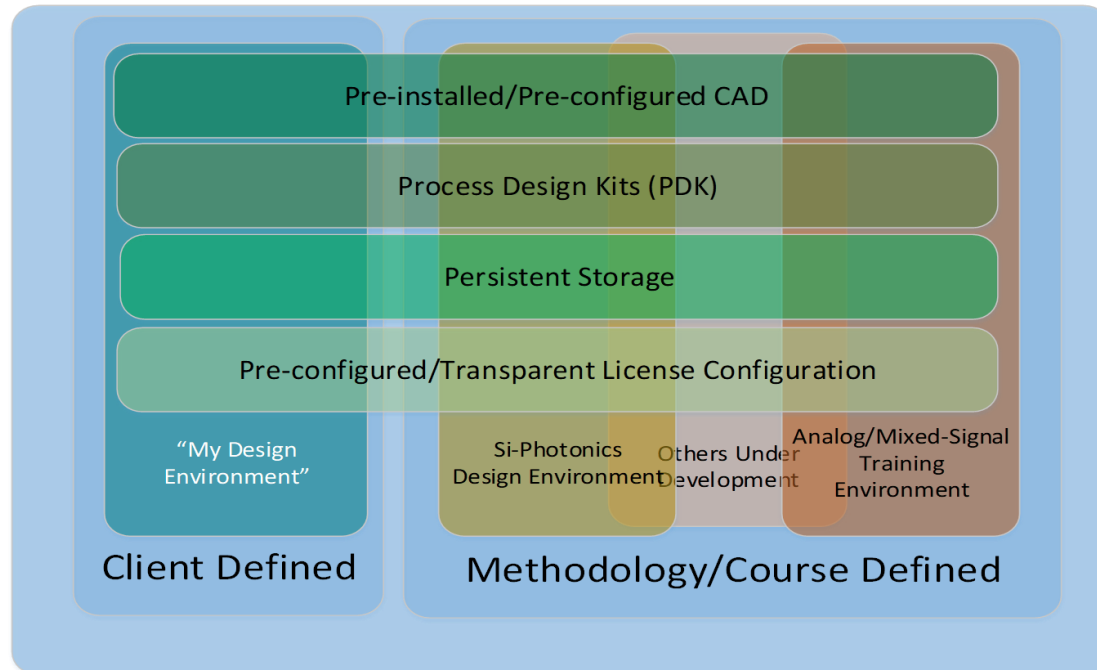
Seamless Transition Between Environments

- **CAD** - Design using CMC Cloud desktop
- **FAB** - Simulate on the CAD Compute cluster
- **LAB** - Prototype on the FPGA+GPU cluster

CMC Cloud: Design Environments



CMC Cloud provides researchers with secure, high-performance, remotely accessible EDA resources for design of advanced microsystems and nanotechnologies.



No local CAD server available?

- Complex design tools (e.g. Cadence, Mentor, Synopsys), scripts and licensing pre-configured and ready

High quality server infrastructure

- Enterprise grade server infrastructure being using to run the tools in CMC Cloud

Time from concept to using tools

- After you discover you need to use a tool, with CMC Cloud you can be fully utilizing the tools within minutes

Immediate access to design flows

- Design flows are **developed** and **supported** by CMC engineers

www.cmc.ca/CMCCloud

CMC Cloud “mini”-HPC Cluster for CAD



Speed up your simulations

- CMC engineers provide assistance in utilizing the infrastructure as well as domain knowledge on utilizing HPC infrastructure
- Documentation/reference designs available for ANSYS, COMSOL, Xilinx and more
- Uniform array available in standard and large memory configurations



CAD Compute Cluster – 8 nodes

- Dual 16-core 2.1-.3.7 GHz CPU
- 4 nodes each with 384GB RAM
- 4 nodes each with 768GB RAM
- 300GB local storage
- 100Gb EDR node interconnect / 10GbE storage

Agenda



- CMC Microsystems
- **CMC Cloud FPGA/GPU Cluster**
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- Use Case : CNN training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

CMC Cloud FPGA/GPU Cluster



CPU, GPU and FPGA in pre-validated cluster to scale heterogenous computing workloads

- CMC engineers provide assistance with access and application best practices
- Hosted and managed by CMC as a cloud resource; accessible at your desktop
- Reference designs using ML as well as heterogeneous computing software stacks

FPGA/GPU Cluster – 8 nodes

- Dual 12 core 2.2-to-3.0 GHz CPU
- 192 GB RAM
- 300 GB local storage
- 100 Gb EDR node interconnect
- 10 GbE storage network
- Xilinx Alveo U200 FPGA
- NVIDIA V100 GPU



| Config. | # of nodes |
|---------------------|------------|
| 2 x GPU | 3 |
| 1 x GPU 1 x FPGA | 2 |
| 2 x FPGA | 3 |

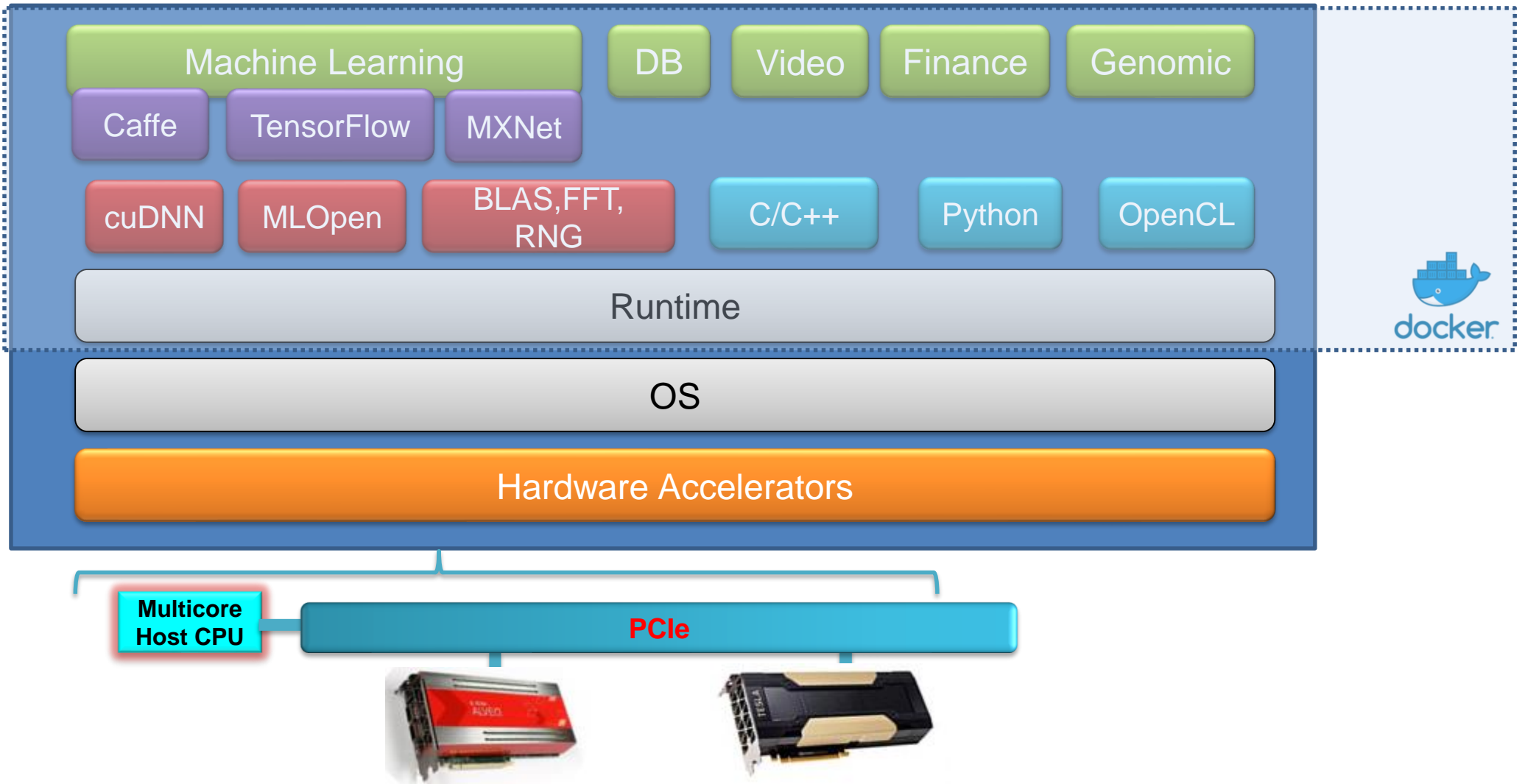
Software stack for the FPGA/GPU cluster

Applications

ML Framework

Middleware,
Tools and Libraries

Hardware



FPGA/GPU cluster use cases



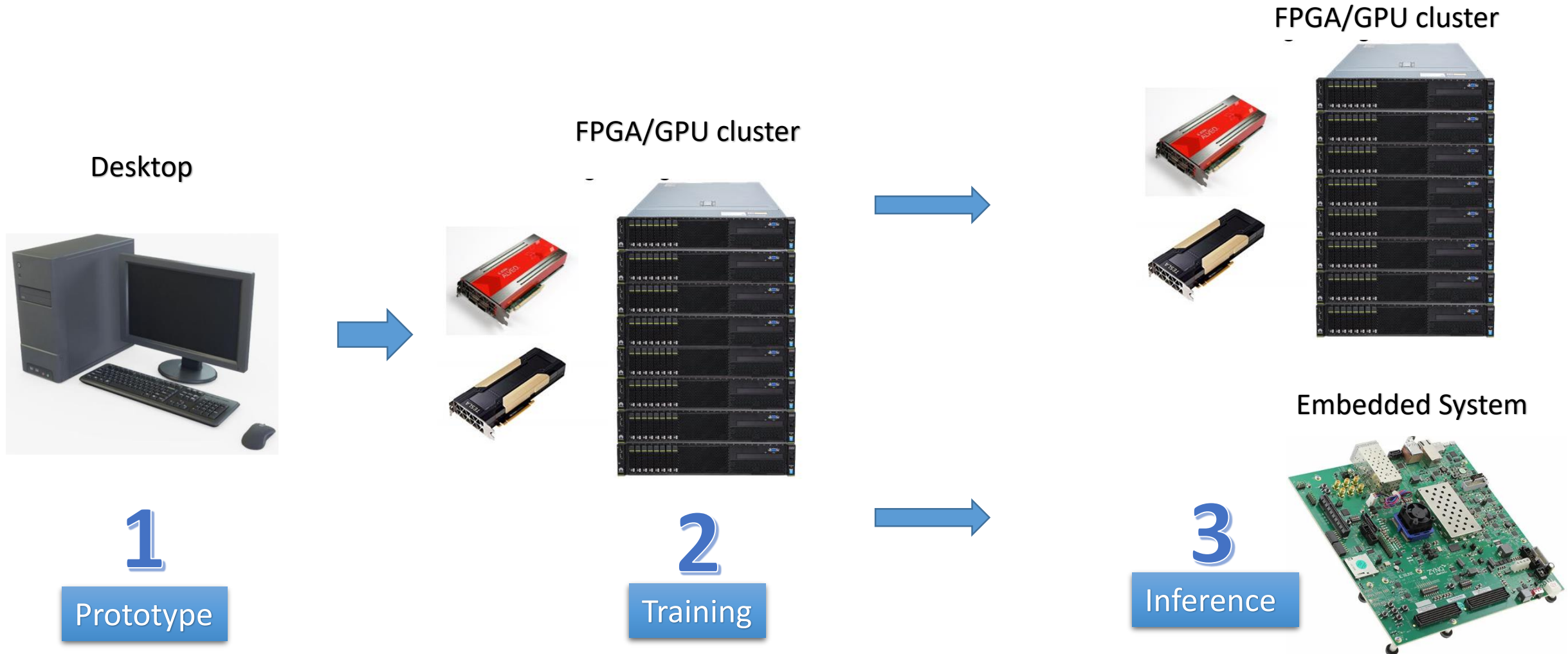
- Machine learning training and inference (e.g. CNN for object detection, speech recognition)
- Quantum chemistry, molecular dynamics, climate and weather, Genomics
- Video Processing / Transcoding
- Financial Computing
- Database analytics
- Networking

Agenda



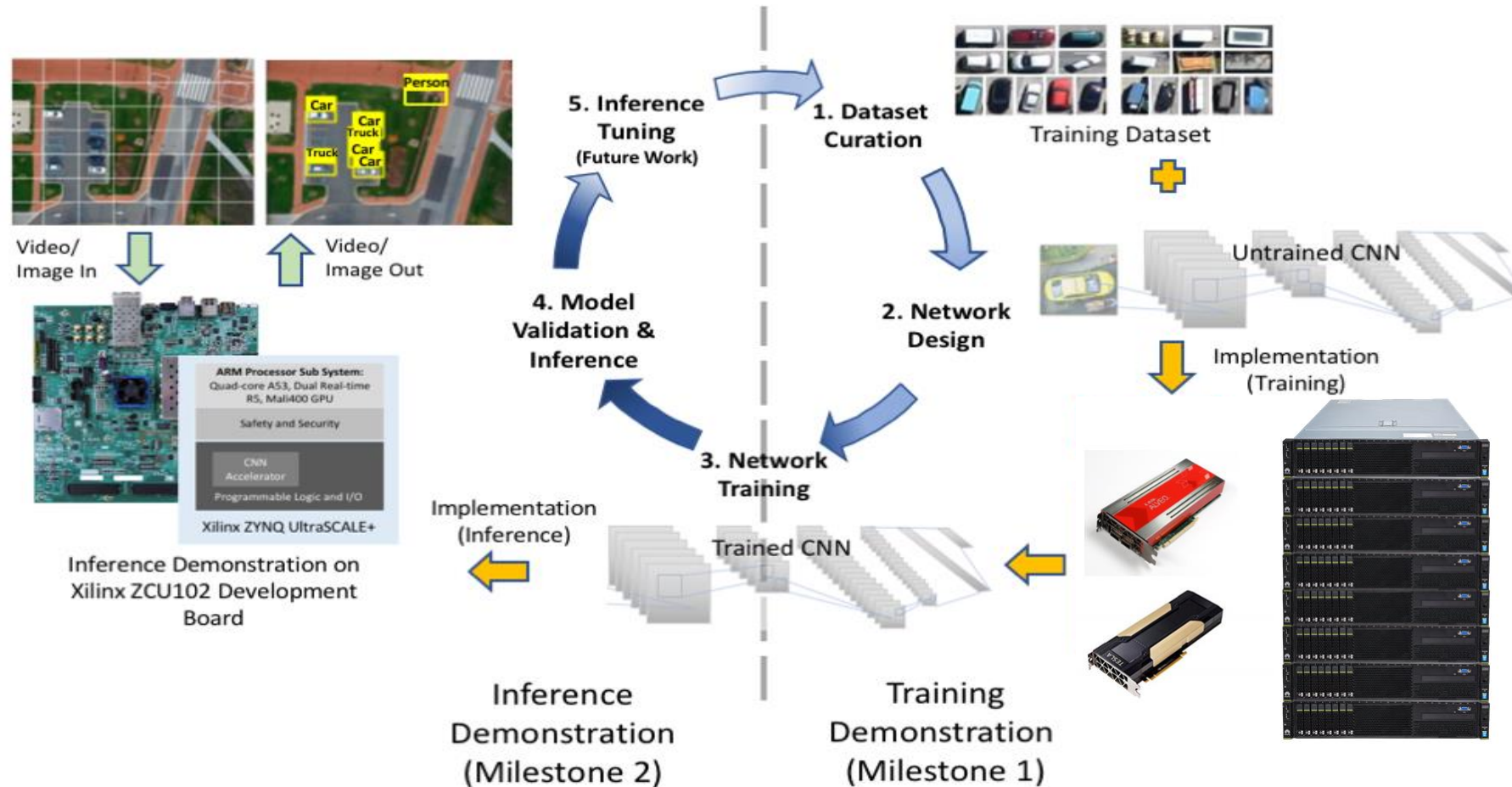
- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- ***End-to-end Deep Learning platform***
- Use Case : CNN training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

End-to-end Deep Learning platform



Innovation for Defence Excellence and Security (IDEaS)

A Novel Platform of Artificial Intelligence-based Object Detection, Classification and Tracking Using Heterogeneous Computing Architectures.



- Two webinars posted on CMC's YouTube channel:
 - *Accelerating Deep Learning for Vision Using Caffe* (February 27, 2019), posted on CMC's YouTube channel
 - *Accelerating Deep Learning for Embedded Vision at the Edge* (May 22, 2019)

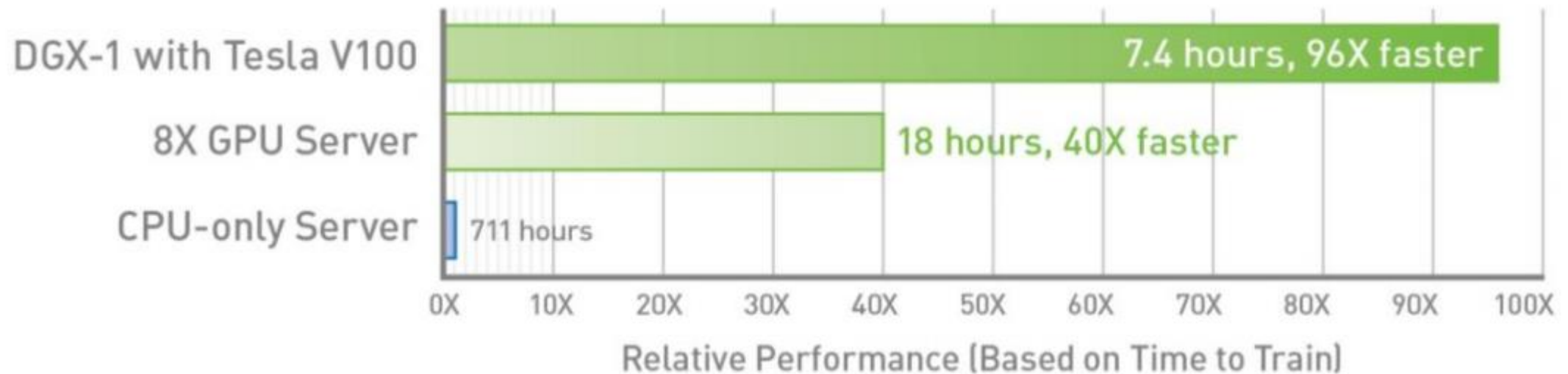
Alveo workloads acceleration



| AREA | PARTNER WORKLOAD | ALVEO ACCELERATION VS CPU |
|--------------------------------|---|---------------------------|
| Database Search and Analytics | BlackLynx Unstructured Data Elasticsearch | 90X |
| Financial Computing | Maxeler Value-at-Risk (VAR) Calculation | 89X |
| Machine Learning | Xilinx Real-Time Machine Learning Inference | 20X |
| Video Processing / Transcoding | NGCodec HEVC Video Encoding | 12X |
| Genomics | Falcon Computing Genome Sequencing | 10X |

Ref. Product Brief Xilinx Alveo U200 & U250

Tesla V100 Acceleration



Workload: ResNet50, 90 epochs to solution | CPU Server: Dual Xeon E5-2699 v4, 2.6GHz

Ref. NVIDIA TESLA V100 GPU ARCHITECTURE

Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- ***Use Case : CNN training implementation using Caffe:***
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

Caffe features

Data pre-processing and management



- **\$CAFFE_ROOT/build/tools**

Data ingest formats

- LevelDB, LMDB database
- HDF5
- Image files

Pre-processing tools

- LevelDB/LMDB creation from raw images
- Generation of the Mean-image
- Training and validation set creation with shuffling

Data transformations

- Image cropping, resizing, scaling and mirroring
- Mean subtraction

Caffe features

Deep Learning model definition



Protobuf model format:

- Developed by Google
- Method of serializing structured data
- Human readable
- Used to define network architecture and training parameters
- No coding required!

```
layer {  
  name: "conv2"  
  type: "Convolution"  
  bottom: "data"  
  top: "conv2"  
  param {  
    lr_mult: 2  
    decay_mult: 0  
  }  
  convolution_param {  
    num_output: 256  
    pad: 2  
    kernel_size: 5  
    group: 2  
    weight_filler {  
      type: "gaussian"  
      std: 0.01  
    }  
    bias_filler {  
      type: "constant"  
      value: 1  
    }  
  }  
}
```


Caffe features

Deep Learning model definition

Loss functions:

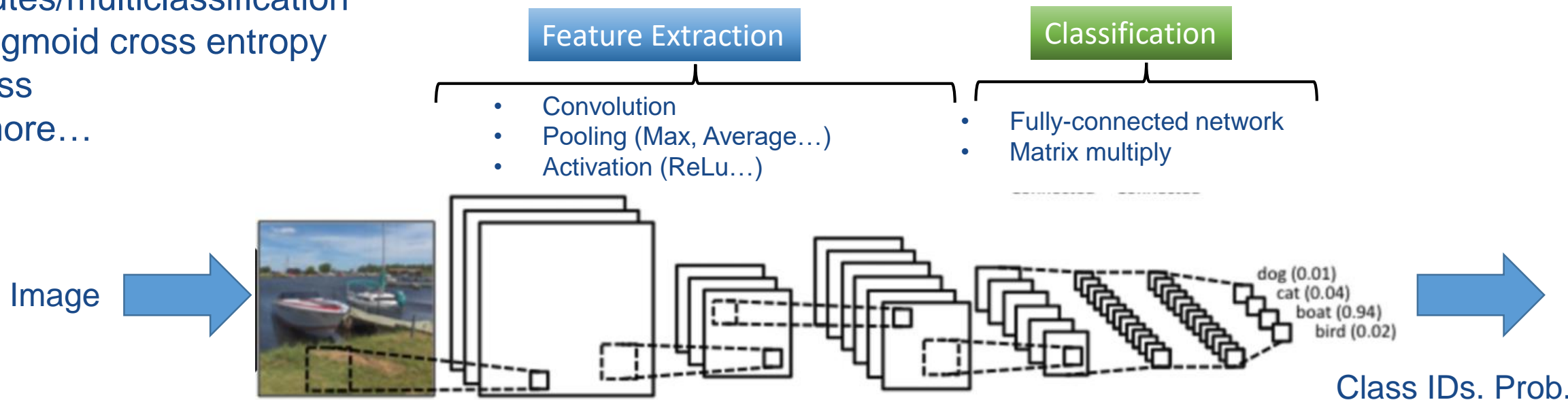
- Classification
 - Softmax
 - Hinge loss
- Linear regression
 - Euclidean loss
- Attributes/multiclassification
 - Sigmoid cross entropy loss
- and more...

Available layer types:

- Convolution
- Pooling
- Normalization
- Data...

Activation functions:

- ReLU
- Sigmoid
- Tanh
- and more...



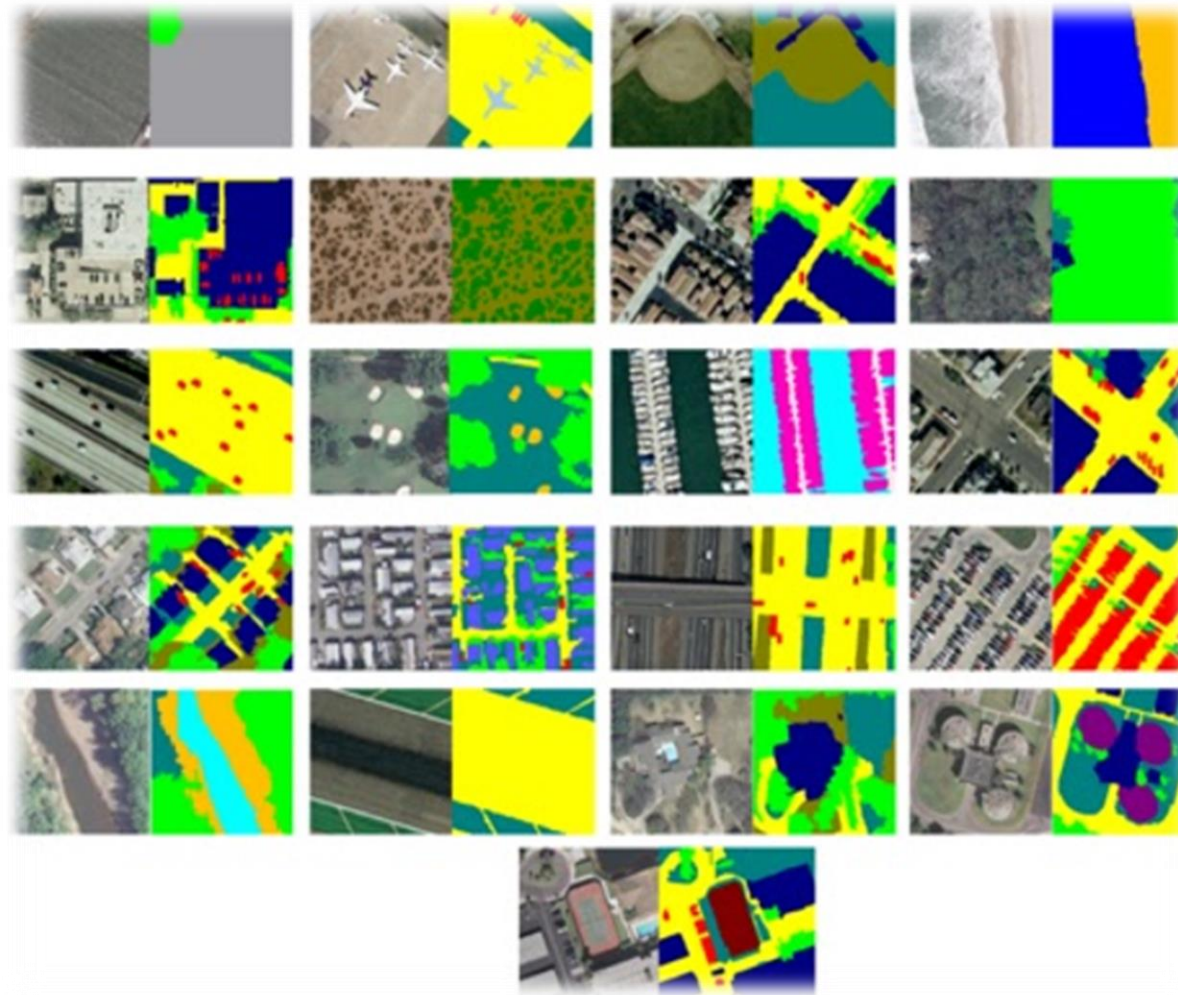
Agenda





- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- ***Use Case : CNN training implementation using Caffe:***
 - **Step 1 - Data preparation**
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

DLRSD dataset

- Extension of the UC Merced archive
- 2100 images 256x256 pixels
- 21 class labels



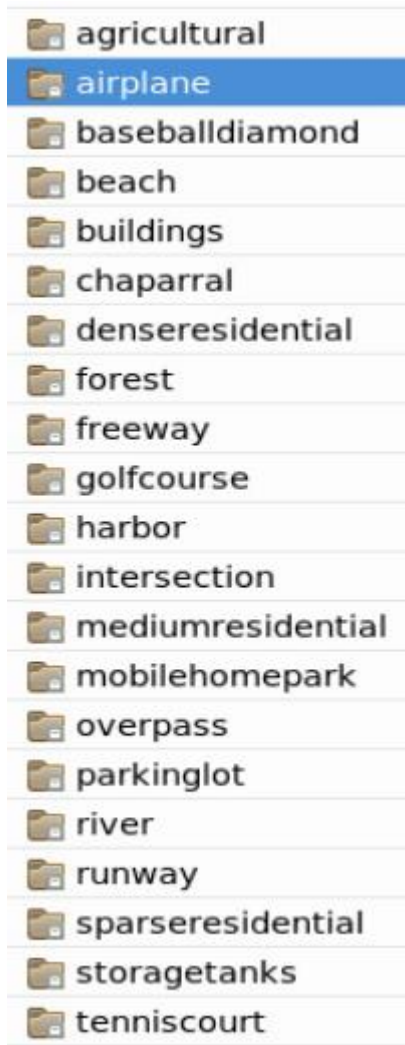
| | |
|---|-------------------|
|  | agricultural |
|  | airplane |
|  | baseballdiamond |
|  | beach |
|  | buildings |
|  | chaparral |
|  | denseresidential |
|  | forest |
|  | freeway |
|  | golfcourse |
|  | harbor |
|  | intersection |
|  | mediumresidential |
|  | mobilehomepark |
|  | overpass |
|  | parkinglot |
|  | river |
|  | runway |
|  | sparseresidential |
|  | storagetanks |
|  | tenniscourt |

Step 1 - Data preparation



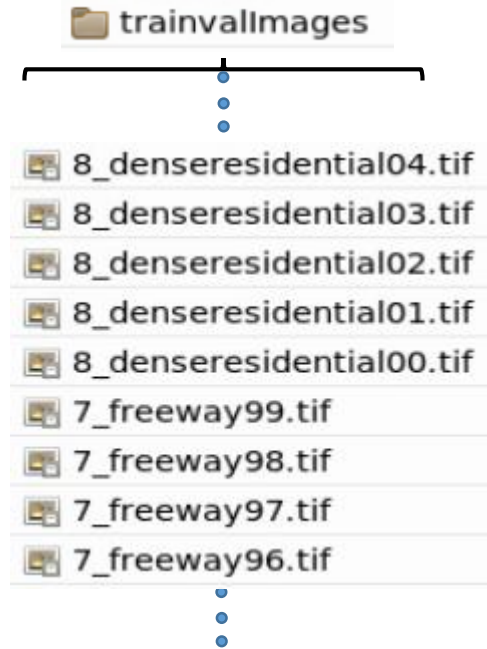
- **Objective**: Create a training and validation databases (from DLRSD dataset) that can be ingested by CAFFE.
- We created two scripts to perform this step:
 - **Script 1**: *prepair_images.py*
 - copy all images from DLRSD directories to one destination directory,
 - creates *train.txt* and *val.txt* required for the training and validation theses text files provide for each image file its class.
 - **Script 2**: *create_dataset_lmdb.sh*
 - resizes all images in the dataset to **227x227** resolution,
 - creates *train_lmdb* as well as *val_lmdb* required for training and validation,
- An additional step in the data preparation is the creation of the mean image *mean.binaryproto* using *make_mean.sh* which is provided by CAFFE.

Step 1 - Data preparation



1

prepair_images.py



| val.txt | train.txt |
|-------------------------------|-----------------------------|
| 4_mobilehomepark62.tif 4 | 18_agricultural04.tif 18 |
| 5_harbor75.tif 5 | 18_agricultural12.tif 18 |
| 1_sparseresidential03.tif 1 | 19_chaparral36.tif 19 |
| 0_overpass73.tif 0 | 11_buildings10.tif 11 |
| 18_agricultural84.tif 18 | 12_tenniscourt69.tif 12 |
| 16_parkinglot47.tif 16 | 3_river20.tif 3 |
| 6_airplane61.tif 6 | 7_freeway51.tif 7 |
| 18_agricultural19.tif 18 | 10_intersection66.tif 10 |
| 10_intersection18.tif 10 | 6_airplane77.tif 6 |
| 17_mediumresidential04.tif 17 | 13_beach97.tif 13 |
| 0_overpass02.tif 0 | 3_river56.tif 3 |
| 15_baseballdiamond45.tif 15 | 14_golfcourse02.tif 14 |
| 9_runway58.tif 9 | 19_chaparral73.tif 19 |
| 19_chaparral89.tif 19 | 11_buildings76.tif 11 |
| 8_denseresidential18.tif 8 | 20_storagetanks01.tif 20 |
| 14_golfcourse73.tif 14 | 10_intersection48.tif 10 |
| 18_agricultural61.tif 18 | 18_agricultural36.tif 18 |
| 9_runway20.tif 9 | 3_river43.tif 3 |
| 14_golfcourse99.tif 14 | 11_buildings26.tif 11 |
| 2_forest80.tif 2 | 2_forest34.tif 2 |
| 4_mobilehomepark66.tif 4 | 8_denseresidential71.tif 8 |
| 19_chaparral94.tif 19 | 20_storagetanks28.tif 20 |
| 17_mediumresidential73.tif 17 | 11_buildings63.tif 11 |
| 3_river41.tif 3 | 11_buildings57.tif 11 |
| 10_intersection13.tif 10 | 5_harbor43.tif 5 |
| 9_runway39.tif 9 | 5_harbor84.tif 5 |
| 9_runway70.tif 9 | 1_sparseresidential48.tif 1 |
| 3_river76.tif 3 | 14_golfcourse06.tif 14 |
| 9_runway67.tif 9 | 2_forest51.tif 2 |
| 18_agricultural75.tif 18 | 8_denseresidential02.tif 8 |
| 17_mediumresidential25.tif 17 | 8_denseresidential84.tif 8 |
| 4_mobilehomepark60.tif 4 | 14_golfcourse97.tif 14 |
| 3_river26.tif 3 | 6_airplane07.tif 6 |
| 5_harbor24.tif 5 | 2_forest66.tif 2 |
| 10_intersection51.tif 10 | 12_tenniscourt85.tif 12 |

2

create_dataset_lmdb.sh



```
GLOG_logtostderr=1 $TOOLS/convert_imageset \
--resize_height=$RESIZE_HEIGHT \
--resize_width=$RESIZE_WIDTH \
--shuffle \
$TRAIN_DATA_ROOT \
$DATA/train.txt \
$EXAMPLE/train_lmdb
```


Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- ***Use Case : CNN training implementation using Caffe:***
 - Step 1 - Data preparation
 - **Step 2 – CNN Model definition**
 - Step 3 - Solver definition
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

Step 2 - Model definition



- Select a CNN architecture and define its parameters in a configuration file *caffenet_train_val_1.prototxt*.
- In this demo, we will use the *bvlc_reference_caffenet* model, which is a replication of *AlexNet*.
- In order to fit this model with the requirement of this project, we need to perform the following modifications:
 - Update the path for input training data, input validation data as well as the path to the mean image.
 - Update the outputs of the fully connected layer “fc8” from 1000 to 21.

Step 2 - Model definition

- *caffenet_train_val_1.prototxt*

1

Change the path for
input data
and mean image

```
name: "CaffeNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    mirror: true
    crop_size: 227
    mean_file: "/home/ideas/.local/install/caffe/cmccideas_dev0/mean.binaryproto"
  }
  data_param {
    source: "/home/ideas/.local/install/caffe/cmccideas_dev0/outlmdb/train_lmdb"
    batch_size: 80
    backend: LMDB
  }
}
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TEST
  }
  transform_param {
    mirror: false
    crop_size: 227
    mean_file: "/home/ideas/.local/install/caffe/cmccideas_dev0/mean.binaryproto"
  }
  data_param {
    source: "/home/ideas/.local/install/caffe/cmccideas_dev0/outlmdb/val_lmdb"
    batch_size: 20
    backend: LMDB
  }
}
```

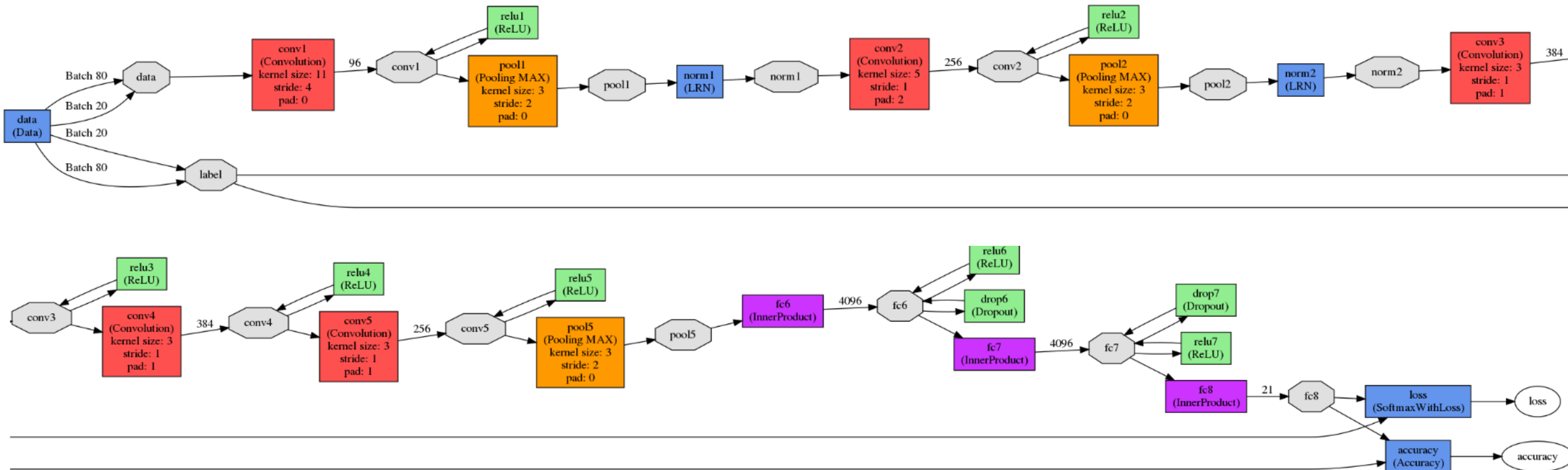
2

Change the
number of outputs
from 1000 to 21

```
type: "InnerProduct"
bottom: "fc7"
top: "fc8"
param {
  lr_mult: 1
  decay_mult: 1
}
param {
  lr_mult: 2
  decay_mult: 0
}
inner_product_param {
  num_output: 21
  weight_filler {
    type: "gaussian"
    std: 0.01
  }
  bias_filler {
    type: "constant"
    value: 0
  }
}
}
layer {
  name: "accuracy"
  type: "Accuracy"
  bottom: "fc8"
  bottom: "label"
  top: "accuracy"
  include {
    phase: TEST
  }
}
layer {
  name: "loss"
  type: "SoftmaxWithLoss"
  bottom: "fc8"
  bottom: "label"
  top: "loss"
}
```

Step 2 - Model definition printing the model

```
> python /home/ideas/.local/install/caffe/ python/draw_net.py  
/home/ideas/.local/install/caffe/cmccideas_dev0/caffenet_train_val_1.prototxt  
/home/ideas/.local/install/caffe/cmccideas_dev0/caffe_model_1.png
```



Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- ***Use Case : CNN training implementation using Caffe:***
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - **Step 3 - Solver definition**
 - Step 4 - Model training
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

Step 3 - Solver definition



- The solver provide parameters to perform model optimisation and guide the training and testing process.
- The content of ***solver_1.prototxt*** is as follow:

```
net: "/home/ideas/.local/install/caffe/cmccideas_dev0/caffenet_train_val_1.prototxt"
test_iter: 400
test_interval: 500
base_lr: 0.001
lr_policy: "step"
gamma: 0.1
stepsize: 5000
display: 20
max_iter: 10000
momentum: 0.9
weight_decay: 0.0005
snapshot: 2000
snapshot_prefix: "/home/ideas/.local/install/caffe/cmccideas_dev0/caffe_model_1"
solver_mode: GPU
```

Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- ***Use Case : CNN training implementation using Caffe:***
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - **Step 4 - Model training**
- Live Demo
 - Training on Tesla V100 GPU
 - Inference on Alveo FPGA
- Q&A

Step 4 - Model training



- At this step, we are ready to train the model by executing the following CAFFE command from the terminal:

```
>caffe train -solver /home/ideas/.local/install/caffe/cmccideas_dev0/solver_1.prototxt 2>&1 | tee /home/ideas/.local/install/caffe/cmccideas_dev0/train.log
```

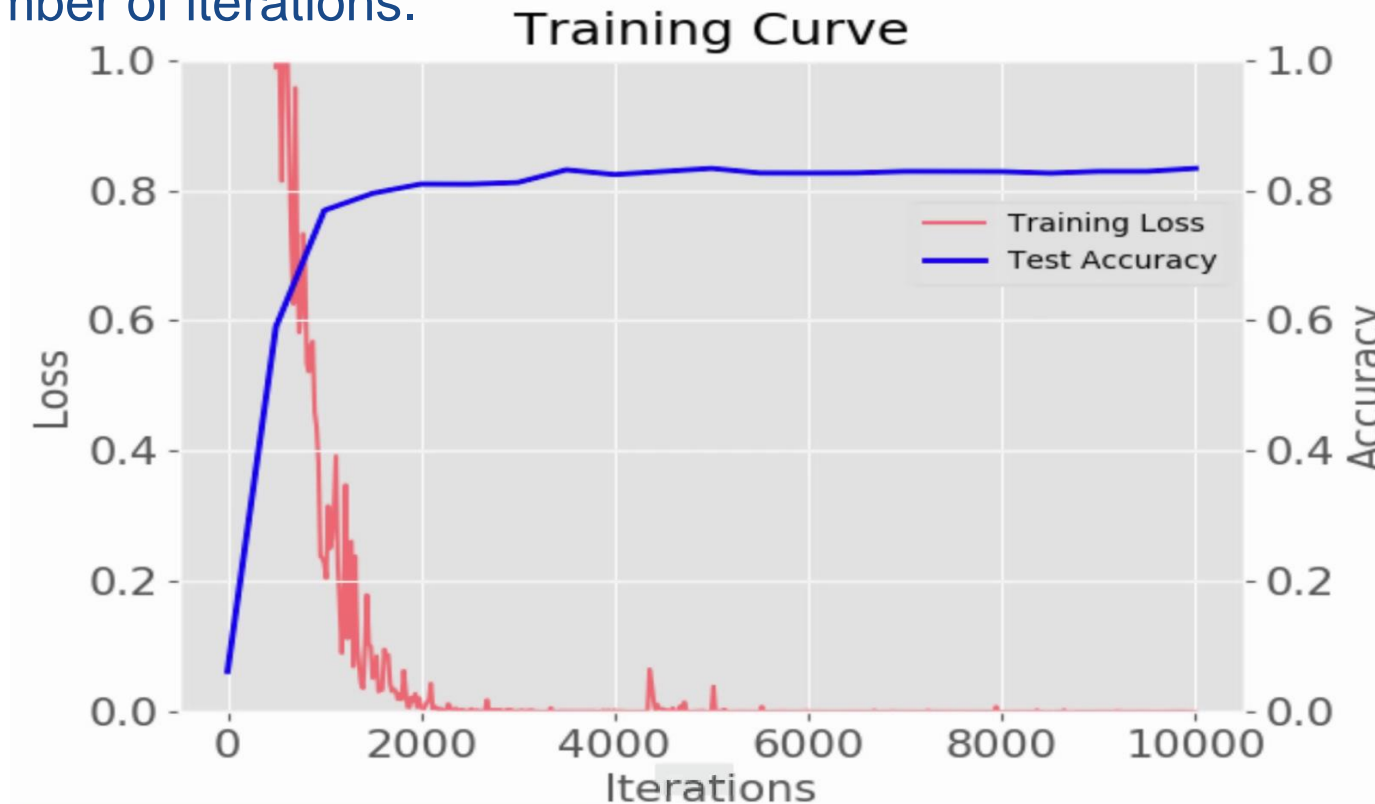
train.log

```
I0205 11:13:50.180753 23320 sgd_solver.cpp:105] Iteration 3900, lr = 0.001
I0205 11:13:50.365981 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:53.088064 23320 solver.cpp:218] Iteration 3920 (6.87914 iter/s, 2.90734s/20 iters), loss = 5.28497e-05
I0205 11:13:53.088107 23320 solver.cpp:237] Train net output #0: loss = 5.27813e-05 (* 1 = 5.27813e-05 loss)
I0205 11:13:53.088116 23320 sgd_solver.cpp:105] Iteration 3920, lr = 0.001
I0205 11:13:53.418174 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:55.995802 23320 solver.cpp:218] Iteration 3940 (6.87827 iter/s, 2.90771s/20 iters), loss = 0.000599943
I0205 11:13:55.995854 23320 solver.cpp:237] Train net output #0: loss = 0.000599875 (* 1 = 0.000599875 loss)
I0205 11:13:55.995863 23320 sgd_solver.cpp:105] Iteration 3940, lr = 0.001
I0205 11:13:56.472354 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:58.904565 23320 solver.cpp:218] Iteration 3960 (6.876 iter/s, 2.90867s/20 iters), loss = 0.000147462
I0205 11:13:58.904662 23320 solver.cpp:237] Train net output #0: loss = 0.000147394 (* 1 = 0.000147394 loss)
I0205 11:13:58.904672 23320 sgd_solver.cpp:105] Iteration 3960, lr = 0.001
I0205 11:13:59.525619 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:14:01.812296 23320 solver.cpp:218] Iteration 3980 (6.87841 iter/s, 2.90765s/20 iters), loss = 0.000356035
I0205 11:14:01.812355 23320 solver.cpp:237] Train net output #0: loss = 0.000355967 (* 1 = 0.000355967 loss)
I0205 11:14:01.812364 23320 sgd_solver.cpp:105] Iteration 3980, lr = 0.001
I0205 11:14:02.579222 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:14:04.524401 23320 solver.cpp:447] Snapshotting to binary proto file /home/ideas/.local/install/caffe/cmccideas_dev0/caffe_model_1_iter_4000.caffemodel
```

```
>python /home/ideas/.local/install/caffe/cmccideas_dev0/plot_learning_curve.py /home/ideas/.local/install/caffe/cmccideas_dev0/train.log /home/ideas/.local/install/caffe/cmccideas_dev0/learning_curve.png
```

Training result

- Figure depicts the resulting learning curve, which is a plot of the training loss and test accuracy as a function of the number of iterations.



- We observe from this figure that the model achieved a validation accuracy of ~85%, and it stopped improving after **4000** iterations.

- **Issues:**
 - CNNs require large datasets and a lot of time to train.
 - Some CNNs could take up to 3-4 weeks to train.
- **Solution:** Transfer learning.
- **Concept:** Instead of training the network from scratch, transfer learning trains an already trained model on a different dataset.
 - **Fine-tune the trained model:**
 - Train the trained model on the new dataset by continuing the backpropagation.
 - We can either fine-tune the whole network or freeze some of its layers.

Model Training with Transfer Learning

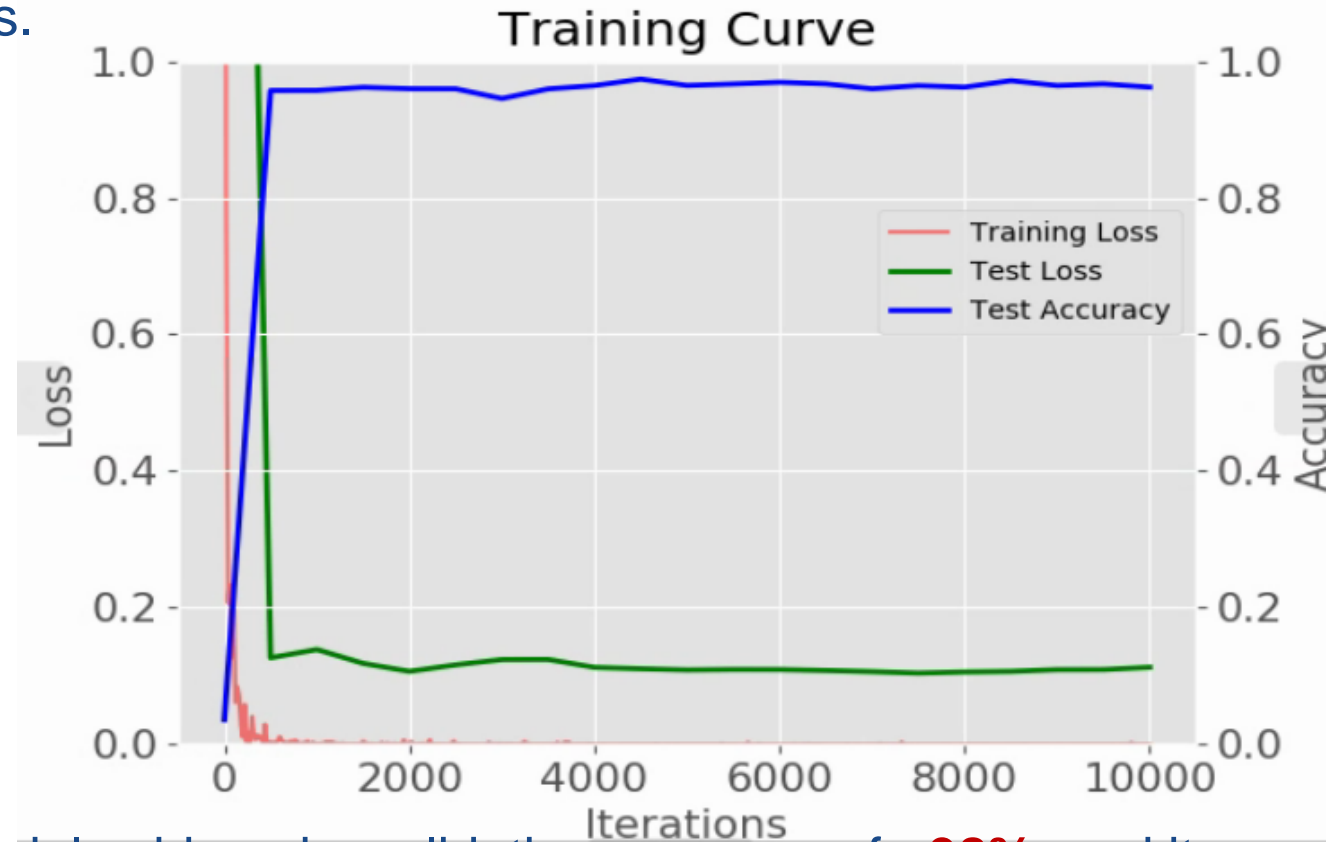


- After defining the model and the solver, we can start training the model by executing the command below.
- Note that we can pass the trained model's weights by using the argument `--weights`

```
> caffe train --solver=/home/ideas/.local/install/caffe/cmccideas_dev0/solver_1.prototxt --weights  
/home/ideas/.local/install/caffe/models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel 2>&1 | tee  
/home/ideas/.local/install/caffe/cmccideas_dev0/train.log
```

Training result

- This figure depicts the resulting learning curve, which is a plot of the training loss and test accuracy as a function of the number of iterations.



- We observe from this figure that the model achieved a validation accuracy of ~98%, and it stopped improving after **1000** iterations.

Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- Use Case : CNN training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- **Live Demo**
 - **Training on Tesla V100 GPU**
 - Inference on Alveo FPGA
- Q&A

Training on the Tesla V100



- `run -it --rm -v /root/scripts/data:/data nvcr.io/nvidia/caffe:19.06-py2`

```
root@uwaccel04:~/scripts# nvidia-docker run -it --rm -v /root/scripts/data:/data nvcr.io/nvidia/caffe:19.06-py2

=====
== NVIDIA Caffe ==
=====

NVIDIA Release 19.06 (build 6768213)
NVIDIA Caffe Version 0.17.3

Container image Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
Copyright (c) 2014, 2015, The Regents of the University of California (Regents)
All rights reserved.

Various files include modifications (c) NVIDIA CORPORATION. All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.

NOTE: Legacy NVIDIA Driver detected. Compatibility mode ENABLED.

NOTE: Detected MOFED driver 4.4-2.0.7; version automatically updated.

NOTE: MOFED driver was detected, but nv_peer_mem driver was not detected.
Multi-node communication performance may be reduced.

NOTE: The SHMEM allocation limit is set to the default of 64MB. This may be
insufficient for NVIDIA Caffe. NVIDIA recommends the use of the following flags:
nvidia-docker run --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 ...

root@bae5457036be:/workspace#
```

Train mnist



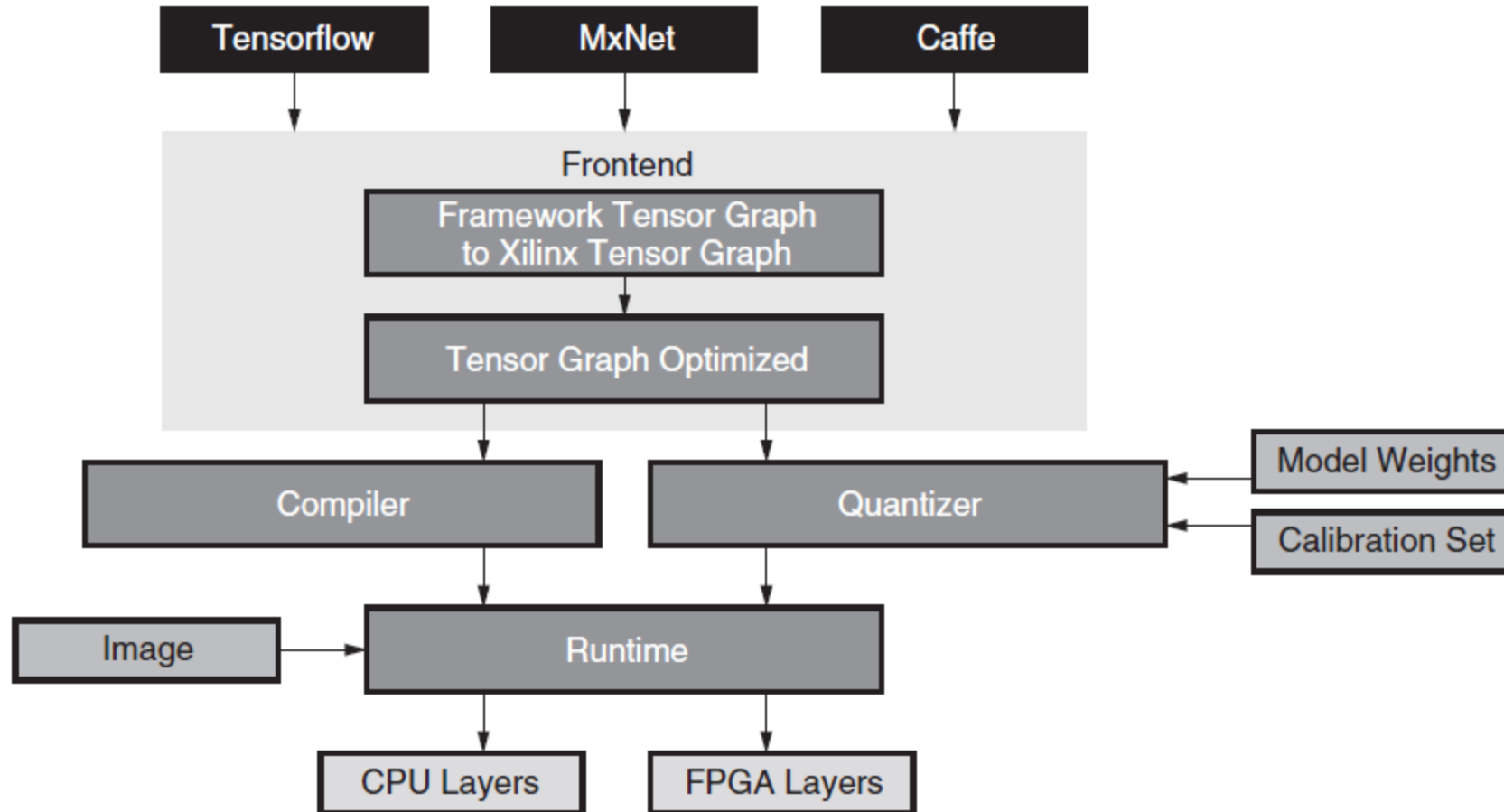
1. Run deviceQuery
2. from /workspace
 - ./data/mnist/get_mnist.sh
 - ./examples/mnist/create_mnist.sh
3. Run training for CPU, then for GPU (examples/mnist/lenet_solver.prototxt)
 - ./examples/mnist/train_lenet.sh

Agenda



- CMC Microsystems
- CMC Cloud FPGA/GPU Cluster
 - HW architecture
 - SW Stack
- End-to-end Deep Learning platform
- Use Case : CNN training implementation using Caffe:
 - Step 1 - Data preparation
 - Step 2 – CNN Model definition
 - Step 3 - Solver definition
 - Step 4 - Model training
- **Live Demo**
 - Training on Tesla V100 GPU
 - **Inference on Alveo FPGA**
- Q&A

xfDNN Software Stack Overview



Ref. Accelerating DNNs with Xilinx Alveo Accelerator Cards

Thank you

Yassine Hariri
CMC Microsystems
Kingston, Ontario, Canada
Hariri@cmc.ca



www.cmc.ca

