Deep Learning/AI Lifecycle with Dell EMC and bitfusion

Bhavesh Patel  
Dell EMC Server Advanced Engineering
Abstract

This talk gives an overview of the end to end application life cycle of deep learning in the enterprise along with numerous use cases and summarizes studies done by Bitfusion and Dell on a high performance heterogeneous elastic rack of DellEMC PowerEdge C4130s with Nvidia GPUs. Some of the use cases that will be talked about in detail will be ability to bring on-demand GPU acceleration beyond the rack across the enterprise with easy attachable elastic GPUs for deep learning development, as well as the creation of a cost effective software defined high performance elastic multi-GPU system combining multiple DellEMC C4130 servers at runtime for deep learning training.
Deep Learning and AI Are being adopted across a wide range of market segments
<table>
<thead>
<tr>
<th>Industry/Function</th>
<th>AI Revolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROBOTICS</td>
<td>Computer Vision &amp; Speech, Drones, Droids</td>
</tr>
<tr>
<td>ENTERTAINMENT</td>
<td>Interactive Virtual &amp; Mixed Reality</td>
</tr>
<tr>
<td>AUTOMOTIVE</td>
<td>Self-Driving Cars, Co-Pilot Advisor</td>
</tr>
<tr>
<td>FINANCE</td>
<td>Predictive Price Analysis, Dynamic Decision Support</td>
</tr>
<tr>
<td>PHARMA</td>
<td>Drug Discovery, Protein Simulation</td>
</tr>
<tr>
<td>HEALTHCARE</td>
<td>Predictive Diagnosis, Wearable Intelligence</td>
</tr>
<tr>
<td>ENERGY</td>
<td>Geo-Seismic Resource Discovery</td>
</tr>
<tr>
<td>EDUCATION</td>
<td>Adaptive Learning Courses</td>
</tr>
<tr>
<td>SALES</td>
<td>Adaptive Product Recommendations</td>
</tr>
<tr>
<td>SUPPLY CHAIN</td>
<td>Dynamic Routing Optimization</td>
</tr>
<tr>
<td>CUSTOMER SERVICE</td>
<td>Bots And Fully-Automated Service</td>
</tr>
<tr>
<td>MAINTENANCE</td>
<td>Dynamic Risk Mitigation And Yield Optimization</td>
</tr>
</tbody>
</table>
...but few people have the time, knowledge, resources to even get started
**PROBLEM 1: HARDWARE INFRASTRUCTURE LIMITATIONS**

- Increased cost with dense servers
- TOR bottleneck, limited scalability
- Limited multi-tenancy on GPU servers (limited CPU and memory per user)
- Limited to 8-GPU applications
- Does not support GPU apps with:
  - High storage, CPU, Memory requirements
PROBLEM 2: SOFTWARE COMPLEXITY OVERLOAD

Software Management
- GPU Driver Management
- Framework & Library Installation
- Deep Learning Framework Configuration
- Package Manager
- Jupyter Server or IDE Setup

Model Management
- Code Version Management
- Hyperparameter Optimization
- Experiment Tracking
- Deployment Automation
- Deployment Continuous Integration

Data Management
- Data Uploader
- Shared Local File System
- Data Volume Management
- Data Integrations & Pipelining

Model Management
- Code Version Management
- Hyperparameter Optimization
- Experiment Tracking
- Deployment Automation
- Deployment Continuous Integration

Workload Management
- Job Scheduler
- Log Management
- User & Group Management
- Inference Autoscaling

Infrastructure Management
- Cloud or Server Orchestration
- GPU Hardware Setup
- GPU Resource Allocation
- Container Orchestration
- Networking Direct Bypass
- MPI / RDMA / RPI / gRPC
- Monitoring

Data Management
- Data Uploader
- Shared Local File System
- Data Volume Management
- Data Integrations & Pipelining

Data Management
- Data Uploader
- Shared Local File System
- Data Volume Management
- Data Integrations & Pipelining

Infrastructure Management
- Cloud or Server Orchestration
- GPU Hardware Setup
- GPU Resource Allocation
- Container Orchestration
- Networking Direct Bypass
- MPI / RDMA / RPI / gRPC
- Monitoring

Workload Management
- Job Scheduler
- Log Management
- User & Group Management
- Inference Autoscaling

Model Management
- Code Version Management
- Hyperparameter Optimization
- Experiment Tracking
- Deployment Automation
- Deployment Continuous Integration

Data Management
- Data Uploader
- Shared Local File System
- Data Volume Management
- Data Integrations & Pipelining
Need to Simplify and Scale
SOLUTION 1/2: CONVERGED RACK SOLUTION

• Up to 64 GPUs per application
• GPU applications with varied storage, memory, CPU requirements
• 30-50% less cost per GPU
• > {cores, memory} / GPU
• >> intra-rack networking bandwidth
• Less inter-rack load
• Composable - Add-as-you-go

Composable compute bundle
**SOLUTION 2/2: COMPLETE, STREAMLINED AI DEVELOPMENT**

1. **DEVELOP**

Develop on pre-installed, quick start deep learning containers.
- Get to work quickly with workspaces with optimized pre-configured drivers, frameworks, libraries, and notebooks.
- Start with CPUs, and attach Elastic GPUs on-demand.
- All your code and data is saved automatically and sharable with others.

2. **TRAIN**

Transition from development to training with multiple GPUs.
- Seamlessly scale out to more GPUs on a shared training cluster to train larger models quickly and cost-effectively.
- Support and manage multiple users, teams, and projects.
- Train multiple models in parallel for massive productivity improvements.

3. **DEPLOY**

Push trained, finalized models into production.
- Deploy a trained neural network into production and perform real-time inference across different hardware.
- Manage multiple AI applications and inference endpoints corresponding to different trained models.
## Dell EMC Deep Learning Optimized servers

### Vertical Segment

### Applications

### Open Source Frameworks

### Optimized Libraries

### Operating System

### Processor/ Accelerator

### Compute Platform

<table>
<thead>
<tr>
<th>Vertical Segment</th>
<th>Applications</th>
<th>Open Source Frameworks</th>
<th>Optimized Libraries</th>
<th>Operating System</th>
<th>Processor/Accelerator</th>
<th>Compute Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell EMC Deep Learning Optimized servers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **DEEP LEARNING FRAMEWORKS**
  - Caffe
  - DL4J
  - Keras
  - CNTK
  - mxnet
  - Theano
  - PyTorch
  - Neon

- **Optimized Libraries**
  - Intel DAAL
  - Intel MKL
  - cuBLAS
  - cuDNN
  - cuPARSE
  - NCCL

- **Operating System**
  - CentOS
  - Linux
  - Ubuntu 16.04 LTS

- **Processor/Accelerator**
  - Intel Xeon
  - Intel Xeon Phi
  - GPU
  - NvLink-GPU
  - KNL Phi in C6320P Sled

- **Compute Platform**
  - C4130
  - R730
  - C6320P in C6300
C4130 DEEP LEARNING Server

Front
- Dual SSD boot drives
- (optional) Redundant Power Supplies
- Front Power Supplies
- IDRAC NIC
- 2x 1Gb NIC

Back
- GPU accelerators (4)
- 8 fans

CPU sockets (under heat sinks)
### GPU DEEP LEARNING RACK SOLUTION

**Features**

- Pre-Built App Containers
- GPU and Workspace Management
- Elastic GPUs across the Datacenter
- Software defined Scaled out GPU Servers

---

#### Configuration Details

<table>
<thead>
<tr>
<th>Features</th>
<th>R730</th>
<th>C4130</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>E5-2669 <a href="mailto:v3@2.1GHz">v3@2.1GHz</a></td>
<td>E5-2630 <a href="mailto:v3@2.4GHz">v3@2.4GHz</a></td>
</tr>
<tr>
<td>Memory</td>
<td>4GB</td>
<td>1TB/node; 64G DIMM</td>
</tr>
<tr>
<td>Storage</td>
<td>Intel PCIe NVME</td>
<td>Intel PCIe NVME</td>
</tr>
<tr>
<td>Networking IO</td>
<td>CX3 FDR InfiniBand</td>
<td>CX3 FDR InfiniBand</td>
</tr>
<tr>
<td>GPU</td>
<td>NA</td>
<td>M40-24GB</td>
</tr>
<tr>
<td>TOR Switch</td>
<td>Mellanox SX6036- FDR Switch</td>
<td></td>
</tr>
<tr>
<td>Cables</td>
<td>FDR 56G DCA Cables</td>
<td></td>
</tr>
</tbody>
</table>
GPU DEEP LEARNING RACK SOLUTION

- Pre-Built App Containers
- GPU and Workspace Management
- Elastic GPUs across the Datacenter
- Software defined Scaled out GPU Servers

End to End Deep Learning Application Life Cycle

1. Develop
2. Train
3. Deploy

GPU Nodes
- C4130 #1
- C4130 #2
- C4130 #3
- C4130 #4

CPU Nodes
- R730 #1
- R730 #2

Infiniband Switch

Optional PSU
PSU
2 SSDs
PCIe 2 x16
PCIe 1 x8
...but wait, ‘converged compute’ requires network attached GPUs...
BITFUSION CORE VIRTUALIZATION

**GPU Device Virtualization**
- Allows dynamic GPU attach on a per-application basis

**Features**
- APIs: CUDA, OpenCL
- Distribution: scale-out to remote GPUs
- Pooling: Oversubscribe GPUs
- Resource Provisioning: Fractional vGPUs
- High Availability: Automatic DMR
- Manageability: Remote nvidia-smi
- *Distributed* CUDA Unified Memory
- Native support for IB, GPUDirect RDMA
- Feature complete with CUDA 8.0
PUTTING IT ALL TOGETHER

Bitfusion Flex, managed containers

CLIENT SERVER

Bitfusion Client Library

GPU SERVER

GPU SERVER

GPU SERVER

Bitfusion Service Daemon
Completely transparent: All CUDA Apps see local and remote GPUs as if directly connected
Results
REMOTE GPUs - LATENCY AND BANDWIDTH

- Data movement overheads is the primary scaling limiter
- Measurements done at application level – cudaMemcpy

<table>
<thead>
<tr>
<th>Native GPUs</th>
<th>Bandwidth Matrix (GB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>H</td>
<td>11.4</td>
</tr>
<tr>
<td>0</td>
<td>94.4</td>
</tr>
<tr>
<td>1</td>
<td>5.5</td>
</tr>
<tr>
<td>2</td>
<td>5.4</td>
</tr>
<tr>
<td>3</td>
<td>5.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Latency Matrix (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>src\ds</td>
</tr>
<tr>
<td>H</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Fast Local GPU copies
PCIe Intranode copies
16 GPU virtual system: Naive implementation w/ TCP/IP

- Fast local GPU copies
- Intranode copies via PCIe
- Low BW, High Latency remote copies

OS Bypass needed to avoid primary TCP/IP overheads
AI apps are very latency sensitive
16 GPU virtual system: Bitfusion optimized transport and runtime

- Same FDR x4 transport, but drop IPoIB
- Replace remote calls with native IB
- Runtime selection of intranode RDMA vs. cudaMemcpy
- Multi-rail communications where available
- Runtime optimizations: pipelining, speculative execution, distributed caching & event coalescing...
Native GPU performance with network attached GPUs

Run time comparison (lower is better)

Multiple ways to create a virtual 4 GPU node, with native efficiency
(secs to train Caffe GoogleNet, batch size: 128)
TRAINING PERFORMANCE

Continued Strong Scaling
Caffe GoogleNet

Weak-scaling
Accelerate Hyper parameter Optimization
Caffe GoogleNet

TensorFlow 1.0 with Pixel-CNN

74% 73% 55% 53% 86%

PCIe host bridge limit

R730

C4130
Other PCIe GPU Configurations Available

Currently Testing

Further reading:

NvLink Configuration

- **4 P100-16GB SXM2 GPU**
- **2 CPU**
- **PCIe switch**
- **1 PCIe slot – EDR IB**
NvLink Configuration

- 4 P100-16GB SXM2 GPU
- 2 CPU
- PCIe switch
- 1 PCIe slot – EDR IB
- Memory: 256GB w/16GB @ 2133
- OS: Ubuntu 16.04
- CUDA: 8.1
Software Solutions
Overview – Bright ML

- Dell EMC has partnered with Bright Computing to offer their Bright ML package as the software stack on Dell EMC Deep learning hardware solution.

- **Open Source Frameworks**
  - TensorFlow, MXNet, CNTK, Theano, Torch, Caffe/Caffe2...

- **Neural Network Libraries**
  - MLPython, CaffeOnSpark, cuDNN, cuBLAS, NCCL, Keras, GIE...

- Bright ML comes with all these Open source frameworks integrated along with the libraries needed to run Neural models.
- It comes as single *.ISO image.

- **Hardware**
  - C4130-NvLink
  - C4130-PCIe GPU

- The Neural networks are trained on hardware like C4130 using NVIDIA GPUs.
### Bright 8.0 Features

- Bright View administrator web interface
- Cloud bursting support for Azure
- New monitoring subsystem
- Ubuntu 16.04 LTS support
- OpenStack Newton
- Mesos integration (+ Marathon)
- Improved Kubernetes integration
- Updated and new machine learning packages
- NVIDIA DCGM integration
- CephFS support
- Job based metrics enabled by default

#### FRAMEWORKS
- Caffe / (Caffe2)
- TensorFlow
- Theano
- Torch
- (CNTK)
- (MXNet)
- (Caffe-MPI)

#### LIBRARIES
- MLPython
- cuDNN
- DIGITS
- CaffeOnSpark
- NCCL
- (GIE)
- (Keras)
Machine Learning in Seismic Imaging Using KNL + FPGA
– Project # 1

Bhavesh Patel - Server Advanced Engineering,
Robert Dildy - Product Technologist Sr. Consultant,
Engineering Solutions
Abstract
This paper is focused on how to apply Machine Learning to seismic imaging with the use of FPGA as a co-accelerator.

It will cover 2 hardware technologies: 1) Intel KNL Phi 2) FPGA and also address how to use Machine learning for seismic imaging.

There are different types of accelerators like GPU, Intel Phi but we are choosing to study how we can use i-ABRA platform on KNL + FPGA to train the neural network using Seismic Imaging data and then doing the inference.

Machine learning in a broader sense can be divided into 2 parts namely : Training and Inference.
Background

Seismic Imaging is a standard data processing technique used in creating an image of subsurface structures of the Earth from measurements recorded at the surface via seismic wave propagations captured from various sound energy sources.

There are certain challenges with Seismic data interpretation like 3D is starting to replace 2D for seismic interpretation.

There has been rapid growth in use of computer vision technology & several companies developing image recognition platforms. This technology is being used for automatic photo tagging and classification. The same concept could be applied to identify geometric patterns in the data and generate image captions/descriptions. We can use Convolutional Neural Networks (CNN) to learn visual concepts using massive amounts of data which would help in doing objective analysis of it.

The use of machine learning and image processing algorithms to analyze, recognize and understand visual content would allow us to analyze data both in Supervised neural networks(SNN) and unsupervised neural networks (UNN) like CNN.
Observing both plane and cross-section

Seismic Stratigraphic image learning

Seismic Geomorphology image learning
Models in plane and cross-section

Seismic Stratigraphic image models

- Flood delta
- Fractured unconformities
- Coastal rock exposure

Train the data to recognize geometrical patterns and utilization of "iPhoto" and "Facebook" technology and methodology to interact with the training.

Seismic Geomorphology image models

- Facies Models
- Algorithms already established in geological modeling software.
- Require some guidance with a low frequency surface model in data to mimic dips and curvatures in stratigraphic response of data.
You give input to the unsupervised training of your data. It will automatically identify similar ones and/or give you a choice of places it finds similar, and you choose to tell its right or wrong.
Solution

For this paper we will be using the following Hardware and Software platforms:

Hardware Platform:
- C6320P Sleds with Intel KNL Phi + Intel Arria 10 (A10PL4) FPGA adapter.

Software Platform:
- i-ABRA Deep learning framework

This will be a joint collaboration with:
- Dell EMC
- Intel
- i-ABRA
- Seismic Imaging firm - TBD
Thank You