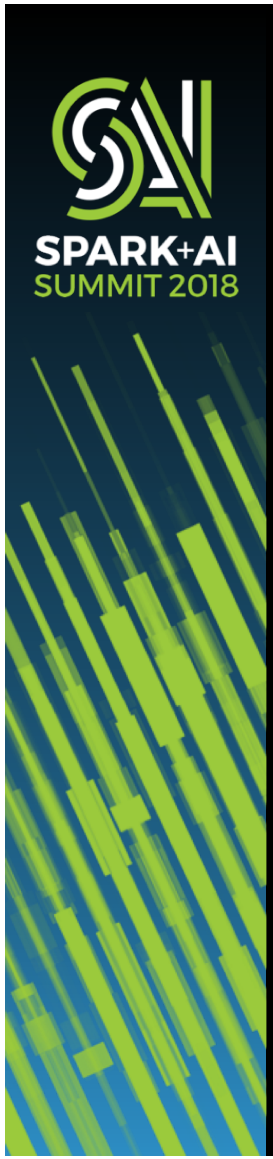


Interactive Deep Learning in Cloud via MMLSpark

Tong Wen, Microsoft

#DL3SAIS

Overview

- Toward a **single environment** for fast experimentation with **big data** and **big compute**

- **Spark** + **Accelerators** (*GPU, FPGA, TPU*, …) + **MPI**
- **High performance** with:
  - **Cost effectiveness**
  - **Ease of use**
  - **Extensibility** and **openness**

MMLSpark

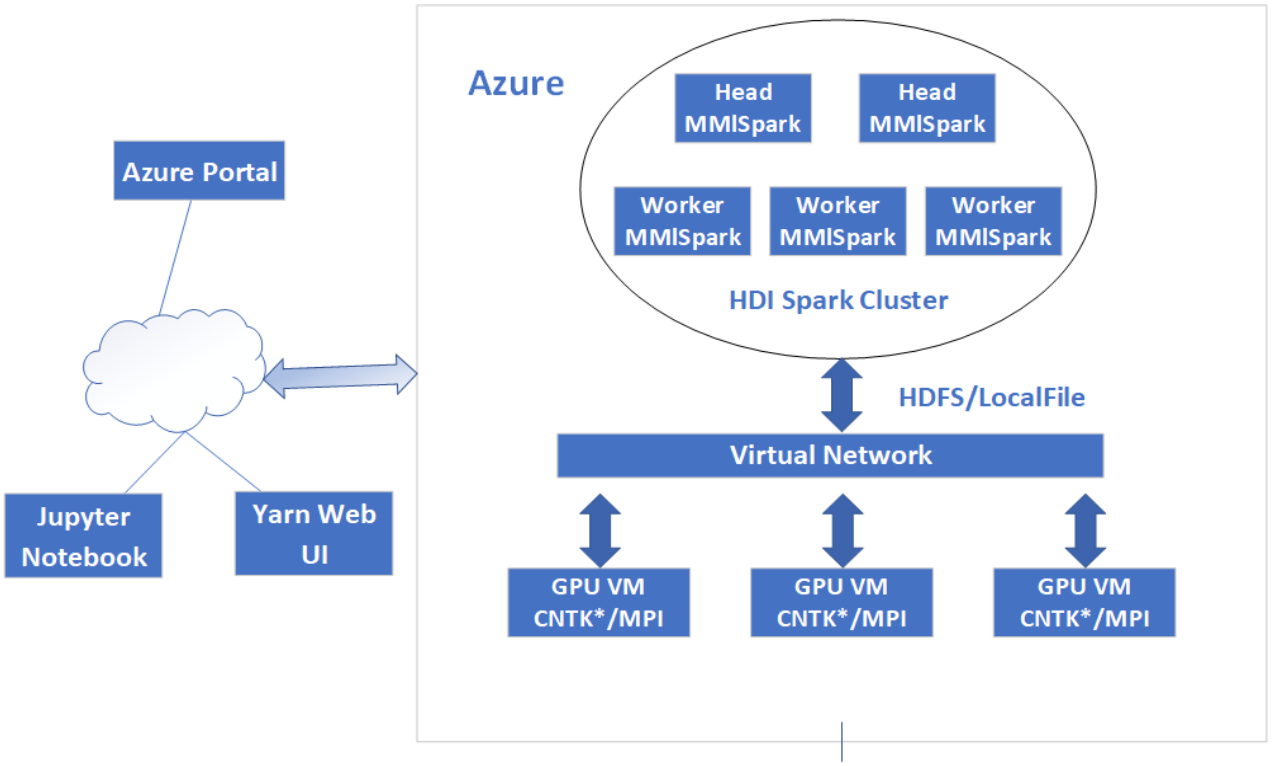
https://github.com/Azure/mmlspark/

- *Tong Wen* *@microsoft.com*
- Akshaya Annavajhala
- Roope Astala
- Eli Barzilay
- Maureen Busch
- Mark Hamilton
- Danil Kirsanov
- Eduardo de Leon
- Ilya Matiach
- Miruna Oprescu
- Young Park
- Sudarshan Raghunathan
- Ratan Sur

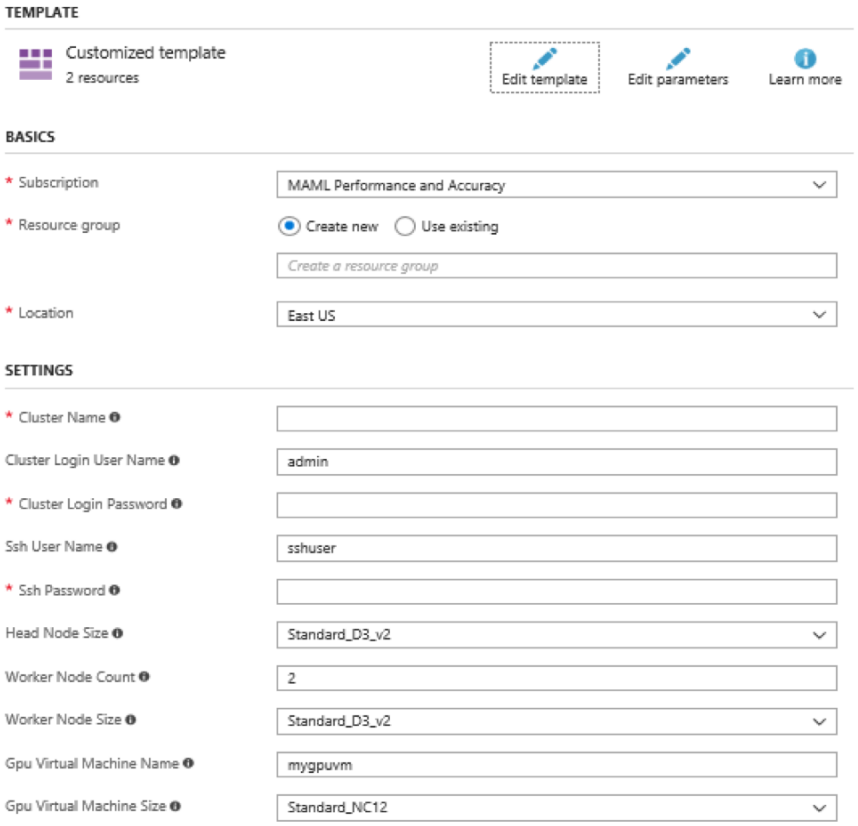
Key Advantages

- **Fast experimentation with Deep Learning**
  - GPU vs CPU: ~***40x*** speedup
  - **Single interactive** environment with easy setup
- **Trained an accurate model on NIH chest X-ray dataset in days**
  - Data size: ***45*** GB compressed on disk; *O(**1**)* TB in memory
  - Model size: ***46*** million parameters
- **Cost to train the above model: < $*9.54***
  - Spark cluster (**10** nodes) **:** $*2.48*/hour
  - **4** GPUs**:** $*2.29*/hour
  - Training time**:** ***54*** mins

Implementation



Setup the System



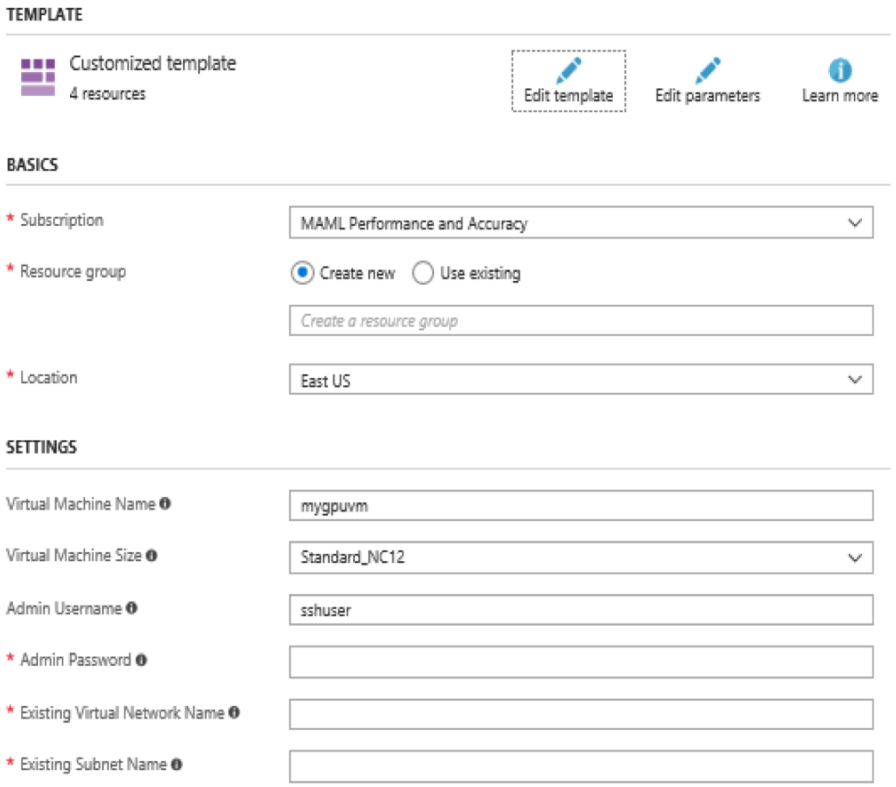
https://github.com/Azure/mmlspark/blob/master/docs/gpu-setup.md

1. Deploy an ARM template within the Azure Portal

→ Click here to open the above main template in the Azure portal.

(If needed, you click the **Edit template** button to view and edit the template.)

Attach a New VM



Set up passwordless SSH login to the GPU VM

```
./setup-ssh-access.sh <vm-name> [<username>]
```

GPU Type	Peak FLOPS/s (FP32)	Price
Tesla K80	**8.7** teraflops	$**0.574**/hour
Tesla P40	**12** teraflops	**1.319**/hour
Tesla P100	**10.6** teraflops	**1.319**/hour
Tesla V100	**15.7** teraflops	**$1.95**/hour
Earth Simulator (2003)	**41** teraflops	>> $*832*/hour

Programming API

In [9]:
```
# Specify the working directory and GPU node name and GPU count
workingDir = "file:/tmp/gpuwork08/"
gpum = ["mygpuvm6,1", "mygpuvm5,1"]
print("Working in " + workingDir)
```

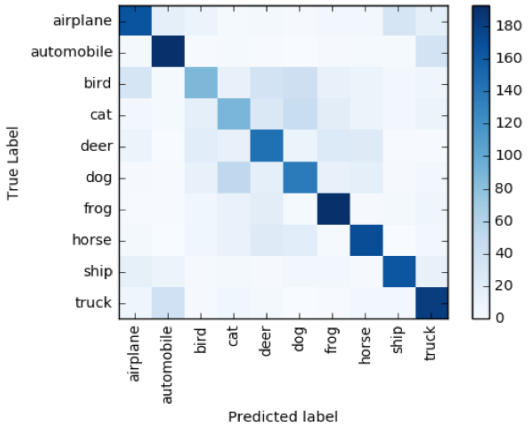
```
Working in file:/tmp/gpuwork08/
```

In [10]:
```
# Train the distributed learner using the VM configured above
learner = CNTKLearner(brainScript=brainscriptText, dataTransfer="hdfs",\
            gpuMachines=gpum, workingDir=workingDir, dataFormat="parquet").fit(trainingSet)
```

In [11]:
```
# Evaluate the model
scoredImages = learner.setOutputNodeName("z").setInputCol("images").setOutputCol("scored").transform(testSet)
scoredImages.show(10)
```

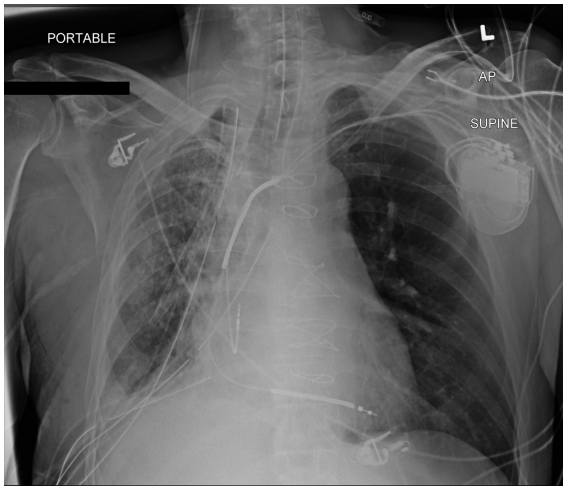
```
+--------------------+--------------------+--------------------+
|              images|              labels|              scored|
+--------------------+--------------------+--------------------+
|[0.0,0.0,3.0,20.0...|[0.0,0.0,0.0,0.0,...|[-1.6580553054809...|
|[1.0,1.0,1.0,1.0,...|[0.0,1.0,0.0,0.0,...|[2.34054183959960...|
|[2.0,0.0,0.0,1.0,...|[0.0,1.0,0.0,0.0,...|[2.14736747741699...|
|[8.0,9.0,10.0,10....|[0.0,0.0,0.0,1.0,...|[-3.2394561767578...|
|[10.0,6.0,15.0,32...|[0.0,0.0,0.0,0.0,...|[-1.3545703887939...|
|[12.0,11.0,13.0,1...|[0.0,1.0,0.0,0.0,...|[3.20597457885742...|
|[12.0,15.0,27.0,3...|[0.0,0.0,0.0,0.0,...|[-1.8197877407073...|
|[13.0,12.0,13.0,1...|[0.0,0.0,0.0,1.0,...|[0.20799629390239...|
|[14.0,44.0,110.0,...|[0.0,0.0,0.0,0.0,...|[-1.5866813659667...|
|[16.0,21.0,21.0,1...|[0.0,0.0,0.0,1.0,...|[-0.1906604766845...|
+--------------------+--------------------+--------------------+
only showing top 10 rows
```

GPU	Epochs	Minibatch size	Wall clock time
Yes	30	32	**1**m**53**s
No	30	32	**73**m**8**s



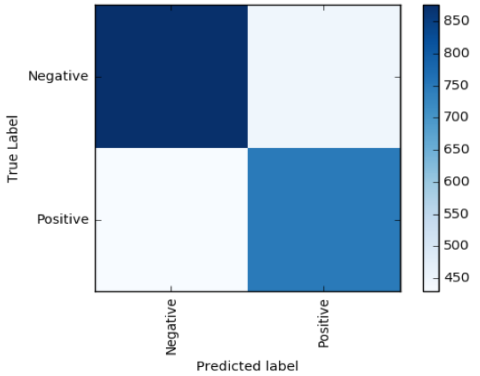
Test Case: NIH Chest X-ray Dataset

- ***112,120*** X-ray images (*1024* by *1024*)
- ***14*** pathology labels
- ***30,805*** unique patients



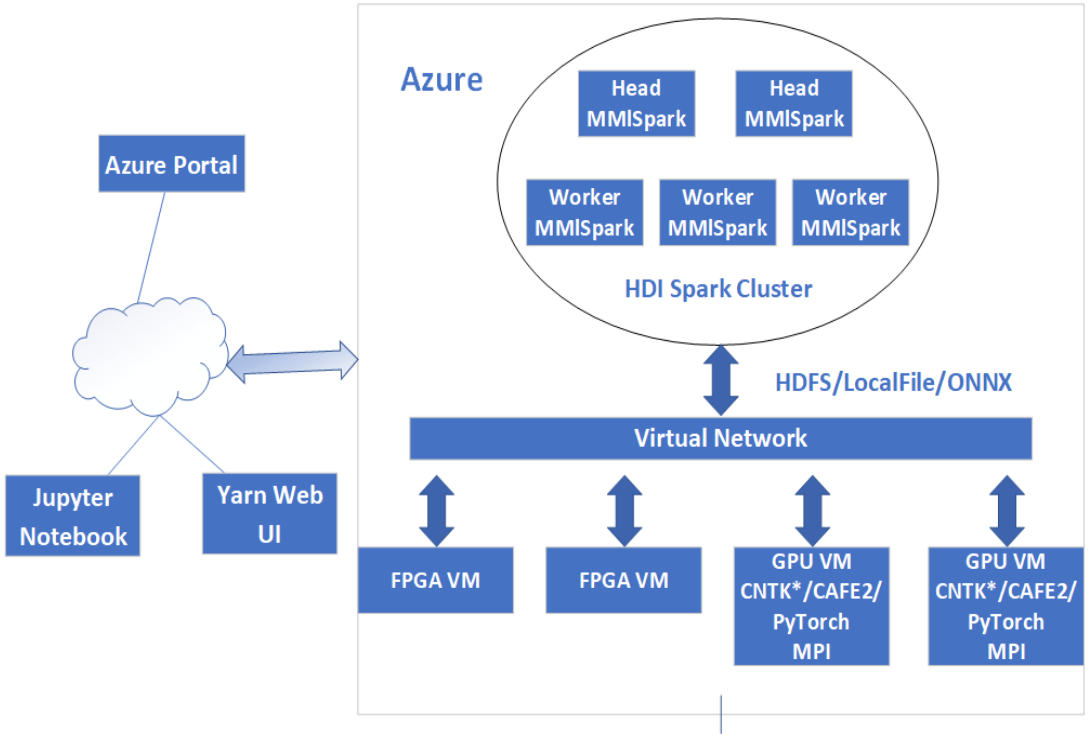
- **AlexNet** with ***46*** million parameters
- **Half of the dataset for training**
- **Downsized to *224* by *224***
- **Binary model**
- **Data Parallel *1-Bit* SGD**

Configuration	Epochs	Minibatch size	Wall clock time
4 GPUs, **2** VMs	55	512	**55**m**47s**
4 GPUs, **4** VMs	55	512	**53**m**40**s



Conclusion & Future Work

- A dynamically **configurable hybrid** architecture to support more ***big data + big compute*** scenarios with **cost effectiveness**
  - **Data exchange** (*Parquet adaptor*)
  - **Model exchange** (*ONNX*)
  - **Single environment** (*Resource management*)
  - **Openness** (*More frameworks*)



Thank You!

https://github.com/Azure/mmlspark/

Test System Configuration

Node Type	Number	Size	Price
Spark Cluster Node	10	**2.4** GHz Intel Xeon® E5-2673 v3 processor; **8** cores; **28**Gib	$**0.248**/hour
GPU VM	2	**1** NVIDIA Tesla **K80** GPU; **6** cores; **56**Gib	$**0.574**/hour
GPU VM	2	**2** NVIDIA Tesla **K80** GPU; **12** cores; **112**Gib	$**1.147**/hour