NVIDIA DEEP LEARNING INSTITUTE

Neural Network Deployment with DIGITS and TensorRT

Twin Karmakharm

Certified Instructor, NVIDIA Deep Learning Institute

DEEP LEARNING INSTITUTE

DLI Mission

Helping people solve challenging problems using AI and deep learning.

- Developers, data scientists and engineers
- Self-driving cars, healthcare and robotics
- Training, optimizing, and deploying deep neural networks

TOPICS

- Caffe
- NVIDIA’S DIGITS
- Deep Learning Approach
- NVIDIA’S TensorRT
- Lab
  - Lab Details
  - Launching the Lab Environment
- Review / Next Steps

CAFFE

Frameworks

Many Deep Learning Tools

. . .

WHAT IS CAFFE?

An open framework for deep learning developed by the Berkeley Vision and Learning Center (BVLC)

caffe.berkeleyvision.org

http://github.com/BVLC/caffe

- Pure C++/CUDA architecture
- Command line, Python, MATLAB interfaces
- Fast, well-tested code
- Pre-processing and deployment tools, reference models and examples
- Image data management
- Seamless GPU acceleration
- Large community of contributors to the open-source project

CAFFE FEATURES

Deep Learning model definition

Protobuf model format

- Strongly typed format
- Human readable
- Auto-generates and checks Caffe code
- Developed by Google
- Used to define network architecture and training parameters
- No coding required!

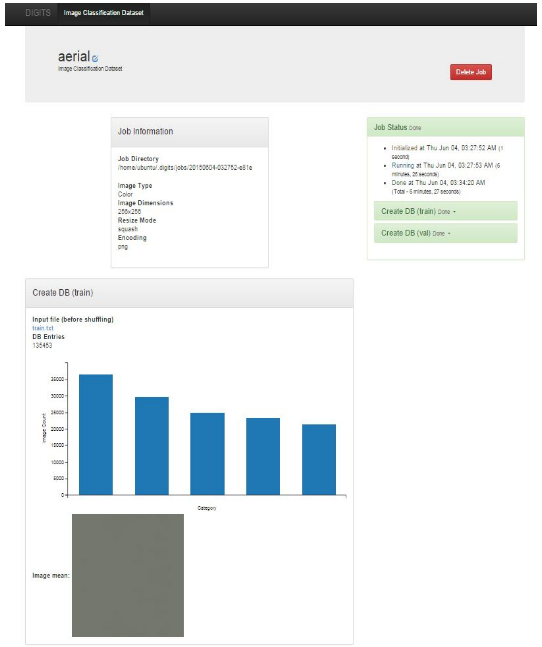
```
name: “conv1”
type: “Convolution”
bottom: “data”
top: “conv1”
convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
        type: “xavier”
    }
}
```

NVIDIA’S DIGITS

NVIDIA’S DIGITS

Interactive Deep Learning GPU Training System

Process Data



aerial
Image Classification Dataset

Job Information

Job Directory
/home/ubuntu/.digits/jobs/20150604-032752-e81e

Image Type
Color
Image Dimensions
256x256
Resize Mode
squash
Encoding
png

Job Status Done

- Initialized at Thu Jun 04, 03:27:52 AM (1 second)
- Running at Thu Jun 04, 03:27:53 AM (6 minutes, 26 seconds)
- Done at Thu Jun 04, 03:34:20 AM (Total - 6 minutes, 27 seconds)

Create DB (train) Done

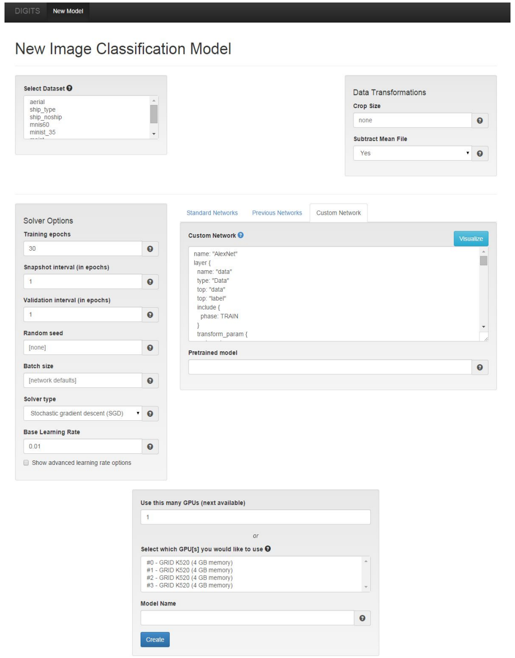
Create DB (val) Done

Create DB (train)

Input file (before shuffling)
train.txt
DB Entries
135453

Image mean:

Configure DNN



New Image Classification Model

Select Dataset
aerial
ship_type
ship_noship
mnis60
minist_35

Data Transformations
Crop Size
none
Subtract Mean File
Yes

Solver Options
Training epochs
30
Snapshot interval (in epochs)
1
Validation interval (in epochs)
1
Random seed
[none]
Batch size
[network defaults]
Solver type
Stochastic gradient descent (SGD)
Base Learning Rate
0.01
□ Show advanced learning rate options

Custom Network

```
name: "AlexNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
```

Pretrained model

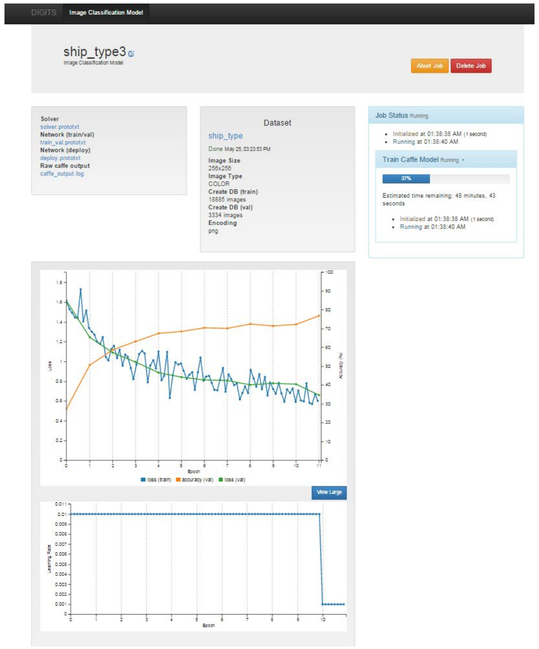
Use this many GPUs (next available)
1

or

Select which GPU[s] you would like to use
#0 - GRID K520 (4 GB memory)
#1 - GRID K520 (4 GB memory)
#2 - GRID K520 (4 GB memory)
#3 - GRID K520 (4 GB memory)

Model Name

Monitor Progress



ship_type3
Image Classification Model

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset
ship_type
Done May 25, 03:23:53 PM
Image Size
256x256
Image Type
COLOR
Create DB (train)
18885 images
Create DB (val)
3334 images
Encoding
png

Job Status Running

- Initialized at 01:38:38 AM (1 second)
- Running at 01:38:40 AM

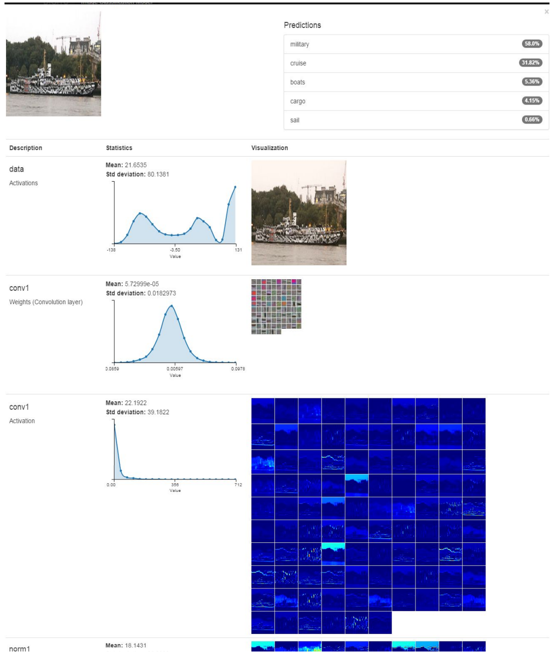
Train Caffe Model Running

37%

Estimated time remaining: 48 minutes, 43 seconds

- Initialized at 01:38:38 AM (1 second)
- Running at 01:38:40 AM

Visualization

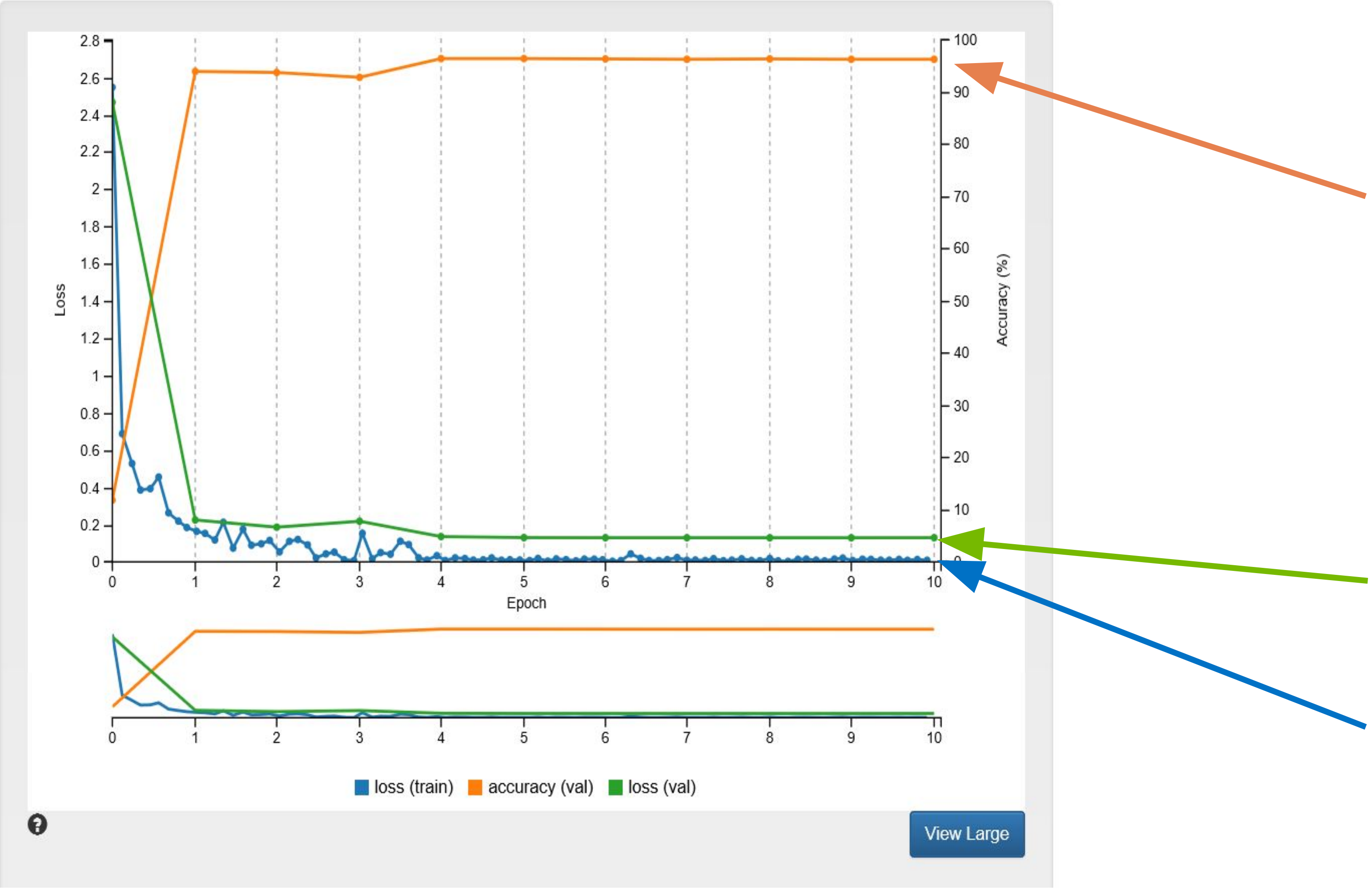


Predictions

military	58.0%
cruise	31.82%
boats	5.36%
cargo	4.15%
sail	0.66%

Description	Statistics	Visualization
data Activations	**Mean:** 21.6535 **Std deviation:** 80.1381	
conv1 Weights (Convolution layer)	**Mean:** 5.72999e-05 **Std deviation:** 0.0182973	
conv1 Activation	**Mean:** 22.1922 **Std deviation:** 39.1822	
norm1	**Mean:** 18.1431	

NVIDIA'S DIGITS



Accuracy obtained from validation dataset

Loss function (Validation)

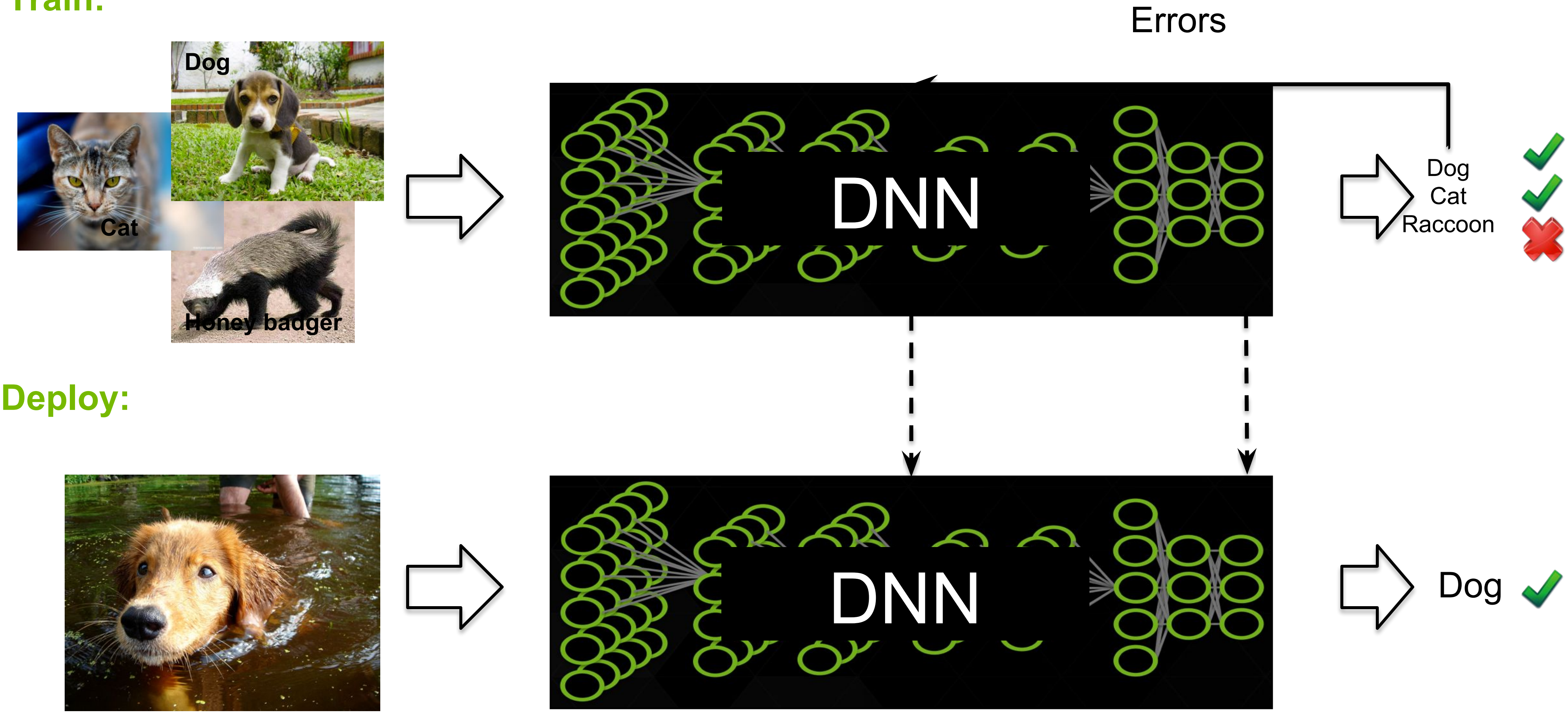
Loss function (Training)

DEEP LEARNING APPROACH

Deep Learning Approach

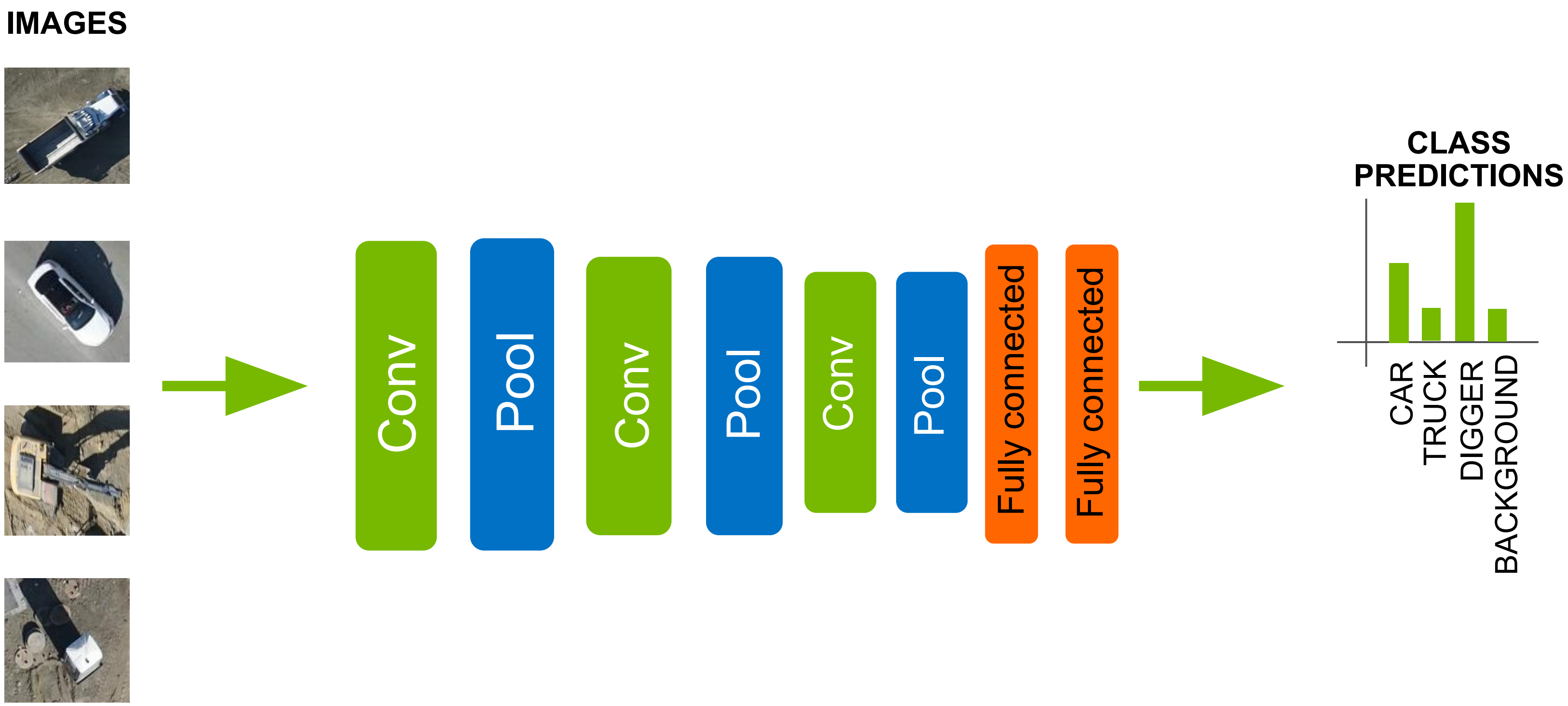
Train:

Deploy:



Deep Learning Approach

Convolutional Neural Network



IMAGES

Conv

Pool

Conv

Pool

Conv

Pool

Fully connected

Fully connected

CLASS PREDICTIONS

CAR

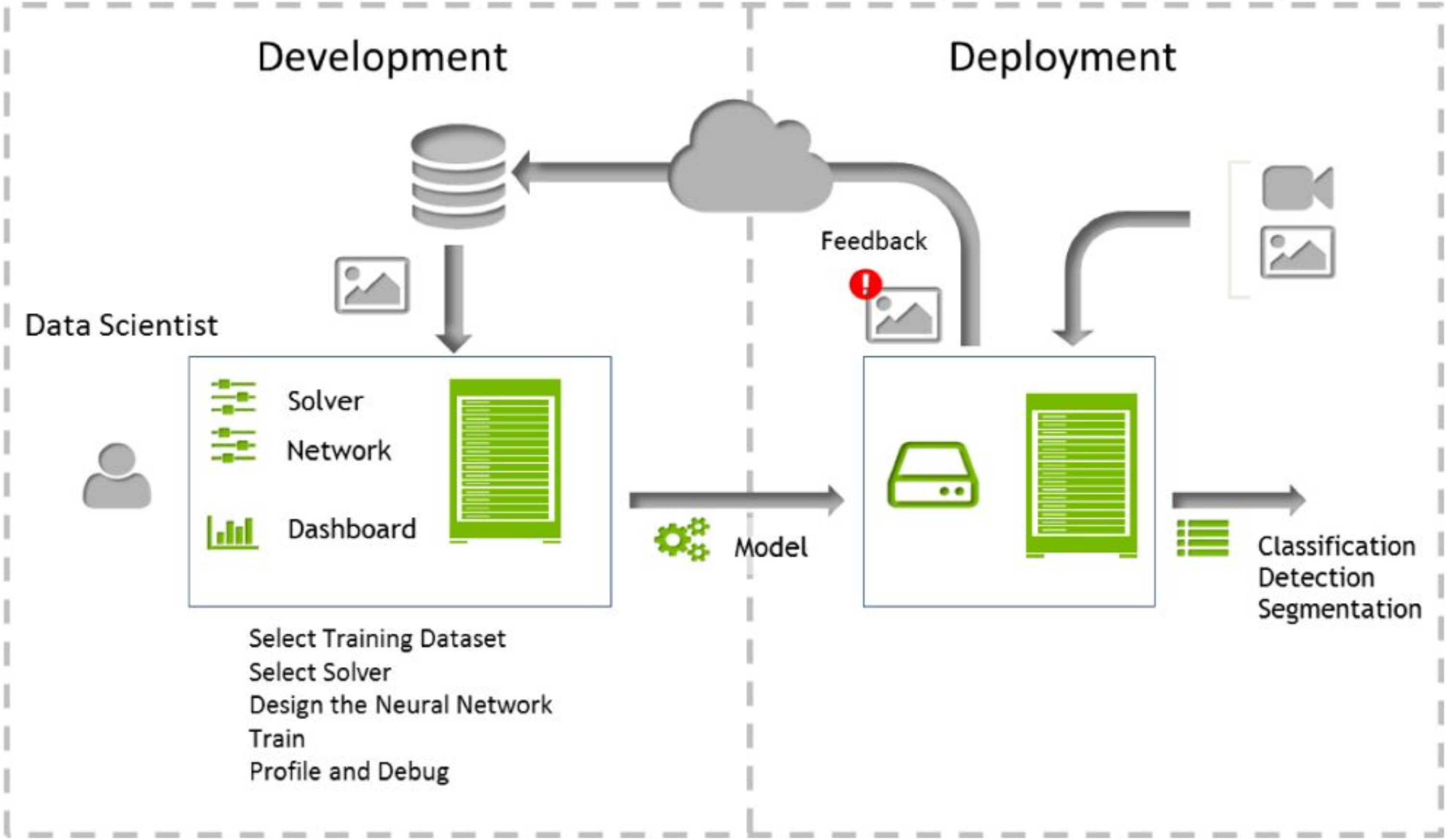
TRUCK

DIGGER

BACKGROUND

Deep Learning Approach

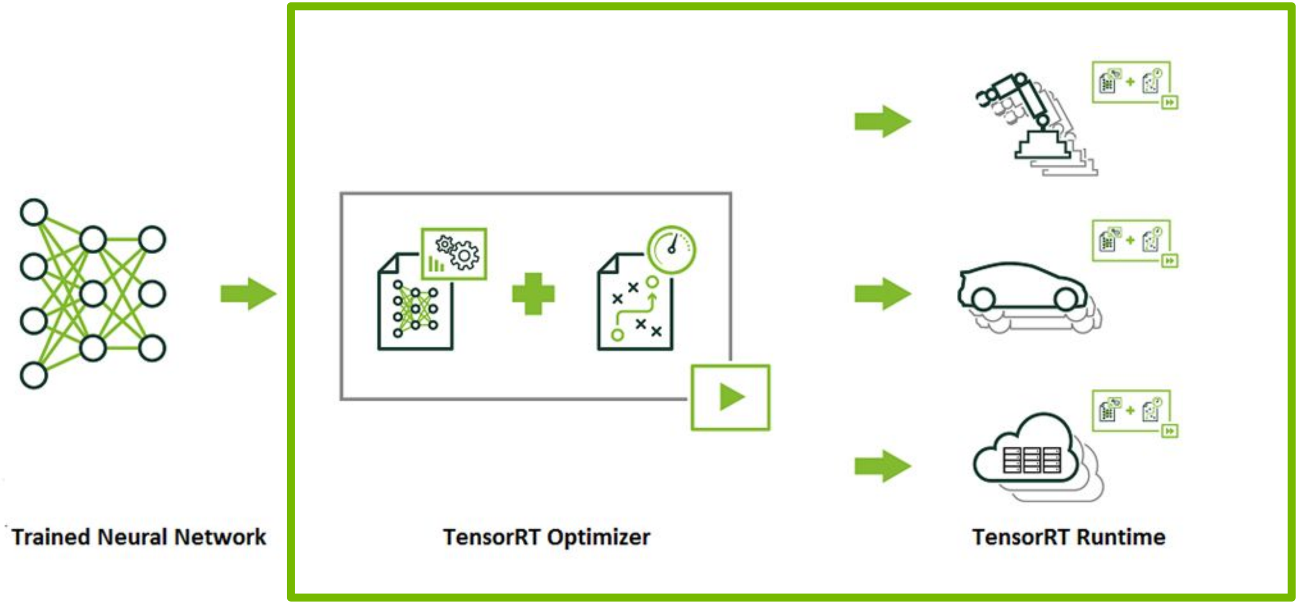
Neural network training and inference



NVIDIA’S TENSORRT

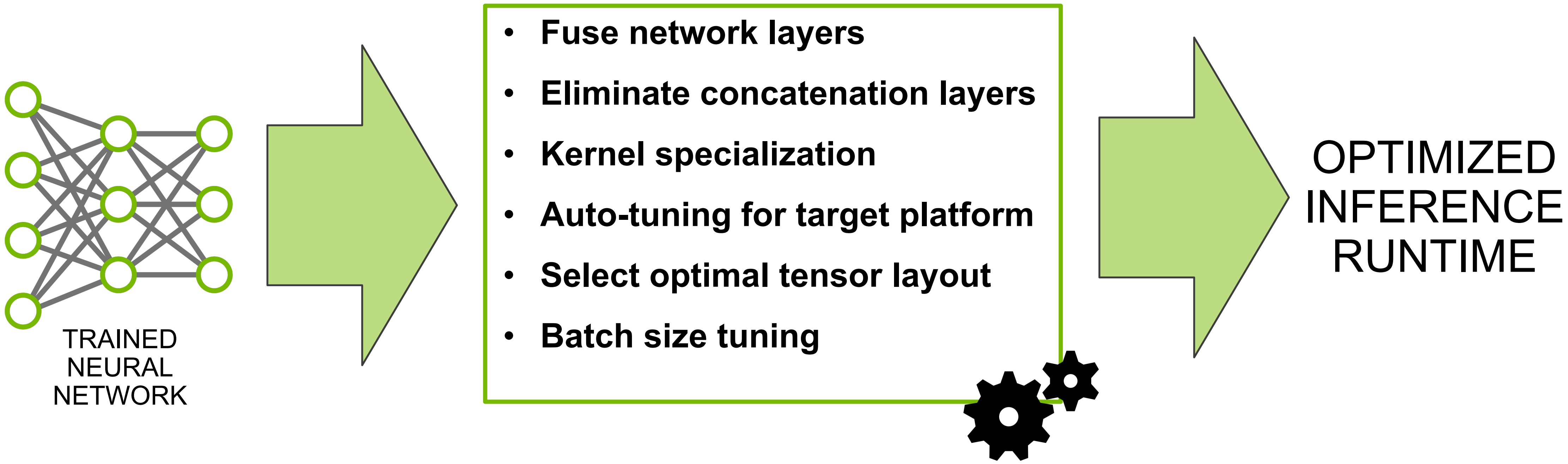
TensorRT

- Inference engine for production deployment of deep learning applications



- Allows developers to focus on developing AI powered applications
  - TensorRT ensures optimal inference performance

TensorRT Optimizer



TRAINED NEURAL NETWORK

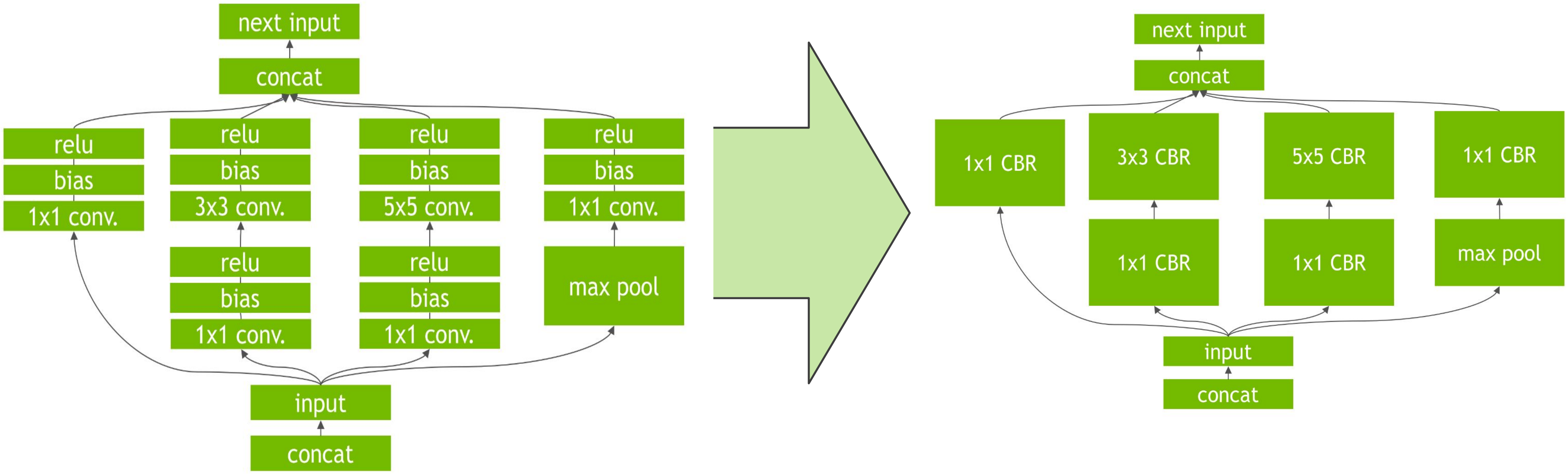
- Fuse network layers
- Eliminate concatenation layers
- Kernel specialization
- Auto-tuning for target platform
- Select optimal tensor layout
- Batch size tuning

OPTIMIZED INFERENCE RUNTIME

developer.nvidia.com/tensorrt

TensorRT Optimizer

Vertical Layer Fusion

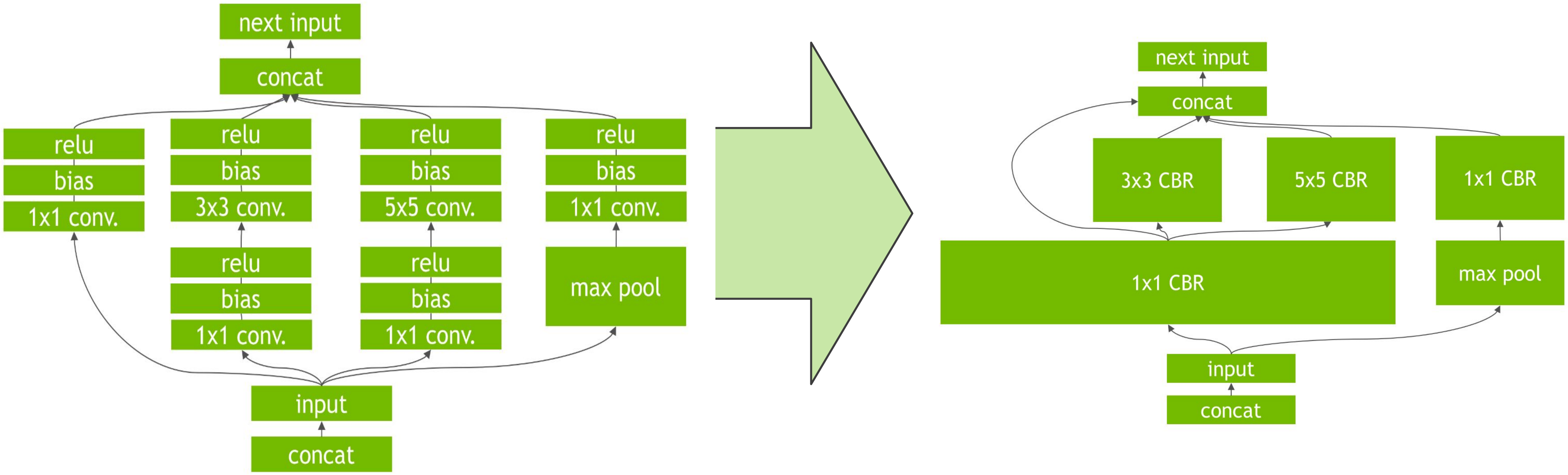


CBR = Convolution, Bias and ReLU

developer.nvidia.com/tensorrt

TensorRT Optimizer

Horizontal Layer Fusion (Layer Aggregation)



CBR = Convolution, Bias and ReLU

developer.nvidia.com/tensorrt

TensorRT Optimizer

Supported layers

- Convolution: 2D
- Activation: ReLU, tanh and sigmoid
- Pooling: max and average
- ElementWise: sum, product or max of two tensors
- LRN: cross-channel only
- Fully-connected: with or without bias
- SoftMax: cross-channel only
- Deconvolution

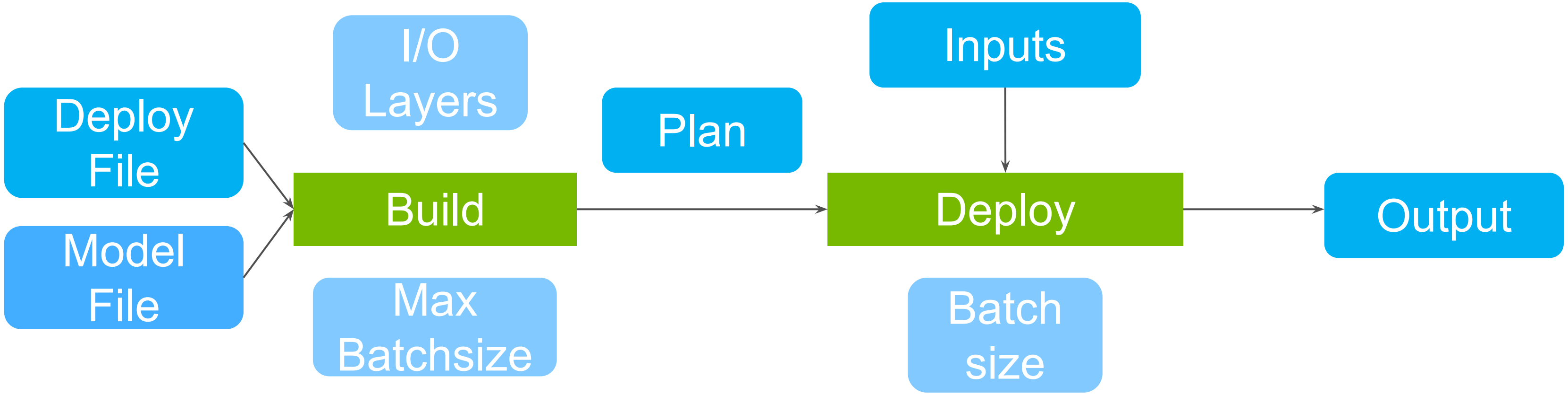
TensorRT Optimizer

- Scalability:
  - Output/Input Layers can connect with other deep learning framework directly
    - Caffe, Theano, Torch, TensorFlow
- Reduced Latency:
  - INT8 or FP16
    - INT8 delivers 3X more throughput compared to FP32
    - INT8 uses 61% less memory compared to FP32

TensorRT Runtime

Two Phases

- **Build:** optimizations on the network configuration and generates an optimized plan for computing the forward pass
- **Deploy:** Forward and output the inference result



TensorRT Runtime

- No need to install and run a deep learning framework on the deployment hardware
- Plan = runtime (serialized) object
  - Plan will be smaller than the combination of model and weights
  - Ready for immediate use
    - Alternatively, state can be serialized and saved to disk or to an object store for distribution
- Three files needed to deploy a classification neural network:
  - Network architecture file (deploy.prototxt)
  - Trained weights (net.caffemodel)
  - Label file to provide a name for each output class

LAB DETAILS

Lab Architectures / Datasets

- *GoogleNet*
  - CNN architecture trained for image classification using the ilsvrc12 Imagenet dataset
  - 1000 class labels to an entire image based on the dominant object present
- *pedestrian_detectNet*
  - CNN architecture able to assign a global classification to an image and detect multiple objects within the image and draw bounding boxes around them
  - Pre-trained model provided has been trained for the task of pedestrian detection using a large dataset of pedestrians in a variety of indoor and outdoor scenes

Lab Tasks

- GPU Inference Engine (GIE) = TensorRT
- Part 1: Inference using DIGITS
  - Will use existing model in DIGITS to perform inference on a single image
- Part 2: Inference using Pycaffe
  - Programming production-like deployable inference code
- Part 3: NVIDIA TensorRT
  - Will run TensorRT Optimizer to build a plan
  - Deploy the plan using TensorRT Runtime

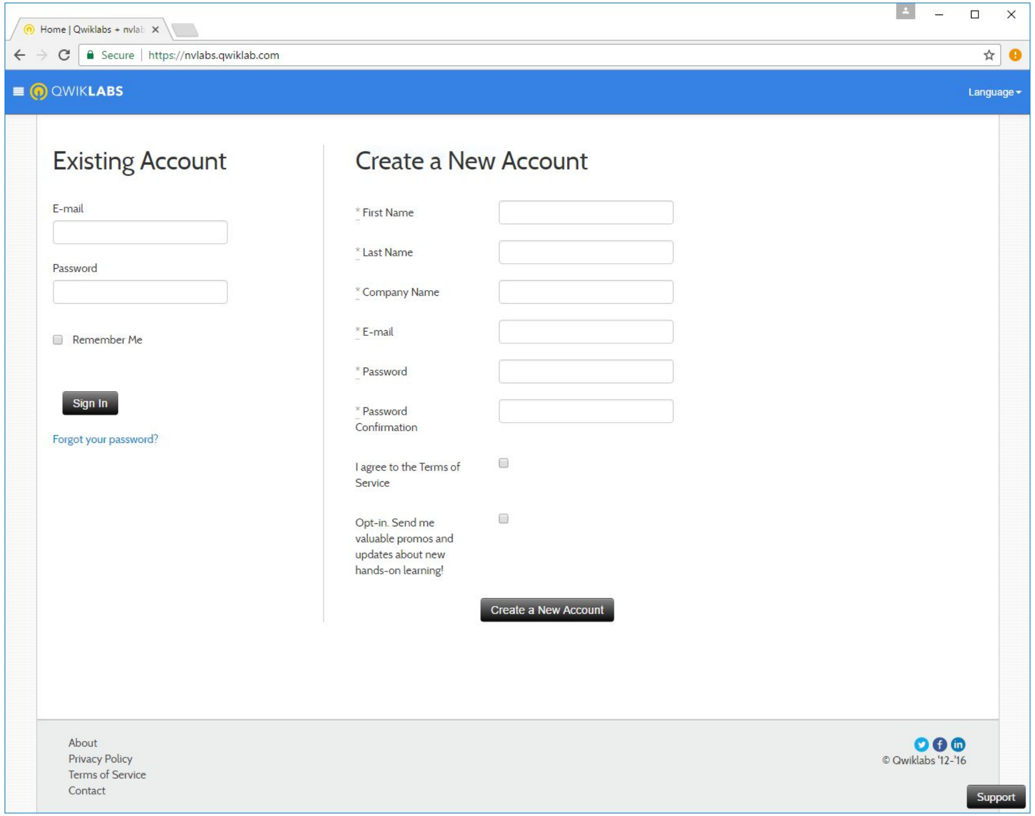
LAUNCHING THE LAB ENVIRONMENT

NAVIGATING TO QWIKLABS

1. Navigate to: https://nvlabs.qwiklab.com

2. Login or create a new account

Please use the email address used to register for session



Existing Account

E-mail

Password

□ Remember Me

Forgot your password?

Create a New Account

* First Name

* Last Name

* Company Name

* E-mail

* Password

* Password Confirmation

I agree to the Terms of Service □

Opt-in. Send me valuable promos and updates about new hands-on learning! □

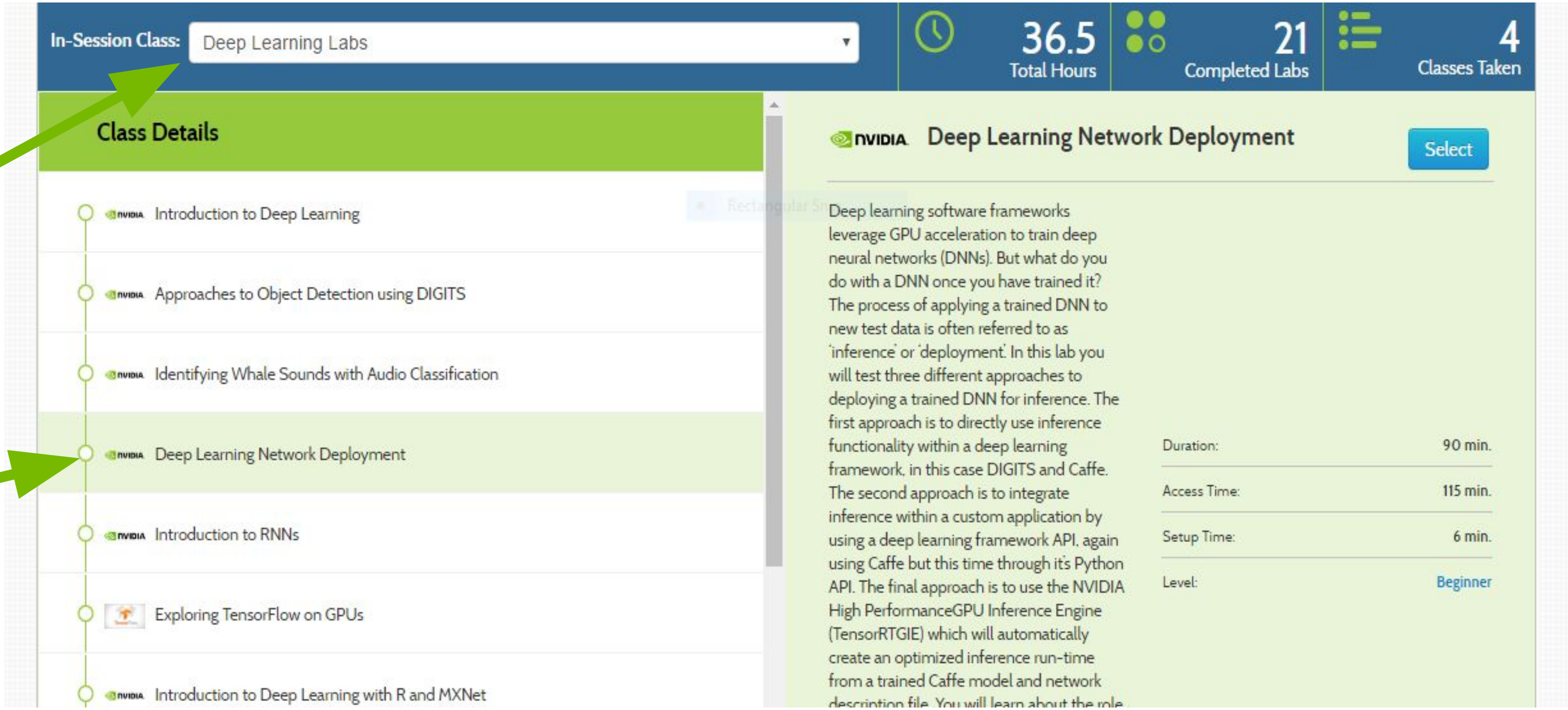
About
Privacy Policy
Terms of Service
Contact

© Qwiklabs '12-'16

ACCESSING LAB ENVIRONMENT

3. Select the event specific In-Session Class in the upper left

4. Click the “Deep Learning Network Deployment” Class from the list



In-Session Class: Deep Learning Labs ▾

36.5 Total Hours | 21 Completed Labs | 4 Classes Taken

Class Details

- Introduction to Deep Learning
- Approaches to Object Detection using DIGITS
- Identifying Whale Sounds with Audio Classification
- Deep Learning Network Deployment
- Introduction to RNNs
- Exploring TensorFlow on GPUs
- Introduction to Deep Learning with R and MXNet

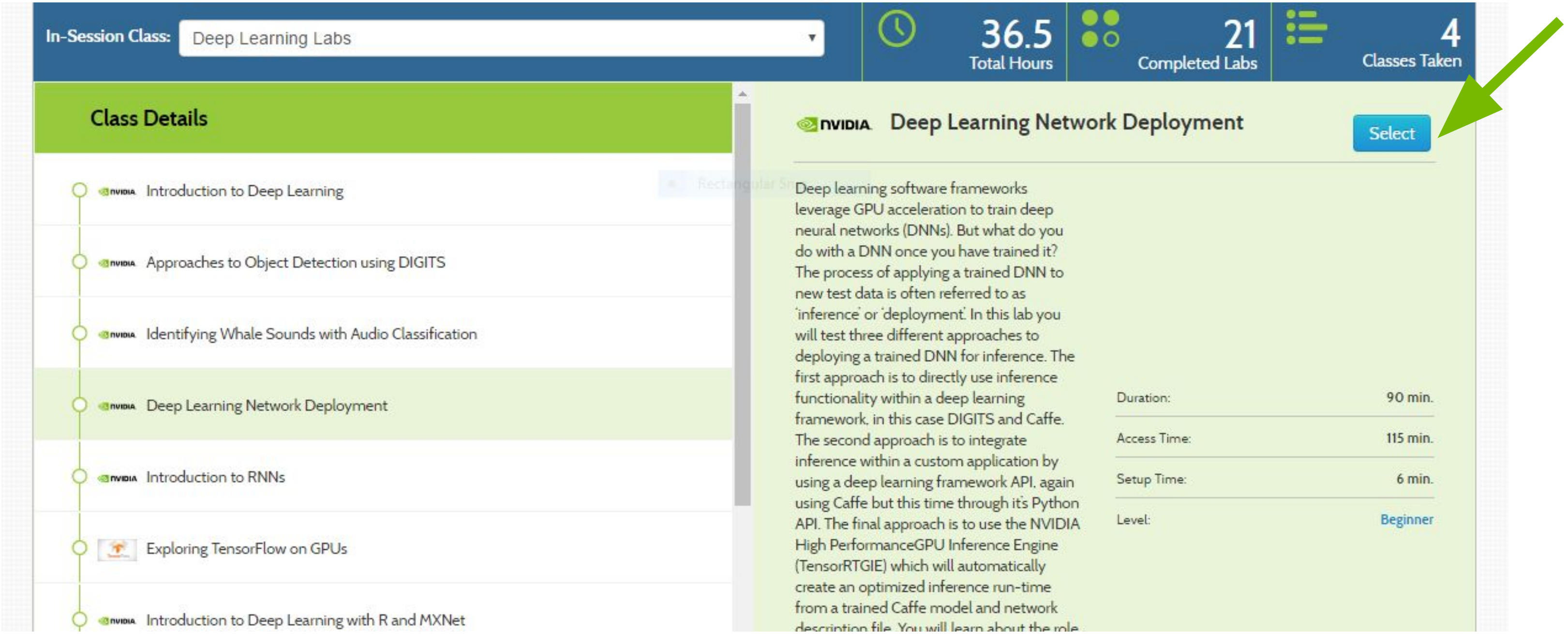
NVIDIA Deep Learning Network Deployment

Select

Deep learning software frameworks leverage GPU acceleration to train deep neural networks (DNNs). But what do you do with a DNN once you have trained it? The process of applying a trained DNN to new test data is often referred to as 'inference' or 'deployment'. In this lab you will test three different approaches to deploying a trained DNN for inference. The first approach is to directly use inference functionality within a deep learning framework, in this case DIGITS and Caffe. The second approach is to integrate inference within a custom application by using a deep learning framework API, again using Caffe but this time through it's Python API. The final approach is to use the NVIDIA High PerformanceGPU Inference Engine (TensorRTGIE) which will automatically create an optimized inference run-time from a trained Caffe model and network description file. You will learn about the role

Duration:	90 min.
Access Time:	115 min.
Setup Time:	6 min.
Level:	Beginner

LAUNCHING THE LAB ENVIRONMENT



Class Details

- Introduction to Deep Learning
- Approaches to Object Detection using DIGITS
- Identifying Whale Sounds with Audio Classification
- Deep Learning Network Deployment
- Introduction to RNNs
- Exploring TensorFlow on GPUs
- Introduction to Deep Learning with R and MXNet

Deep Learning Network Deployment

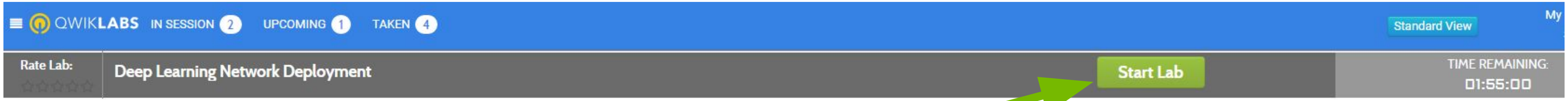
Deep learning software frameworks leverage GPU acceleration to train deep neural networks (DNNs). But what do you do with a DNN once you have trained it? The process of applying a trained DNN to new test data is often referred to as 'inference' or 'deployment'. In this lab you will test three different approaches to deploying a trained DNN for inference. The first approach is to directly use inference functionality within a deep learning framework, in this case DIGITS and Caffe. The second approach is to integrate inference within a custom application by using a deep learning framework API, again using Caffe but this time through it's Python API. The final approach is to use the NVIDIA High PerformanceGPU Inference Engine (TensorRTGIE) which will automatically create an optimized inference run-time from a trained Caffe model and network description file. You will learn about the role

Duration:	90 min.
Access Time:	115 min.
Setup Time:	6 min.
Level:	Beginner

5. Click on the Select button to launch the lab environment

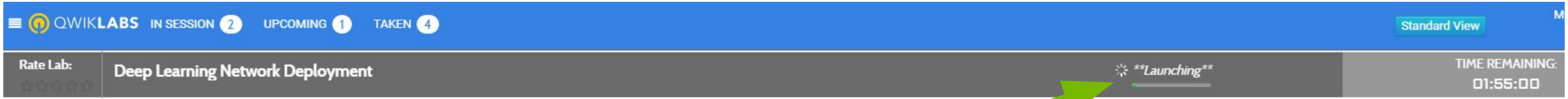
   - After a short wait, lab Connection information will be shown
   - Please ask Lab Assistants for help!

LAUNCHING THE LAB ENVIRONMENT



6. Click on the Start Lab button

LAUNCHING THE LAB ENVIRONMENT



QWIKLABS IN SESSION 2 UPCOMING 1 TAKEN 4

Standard View

M

Rate Lab:

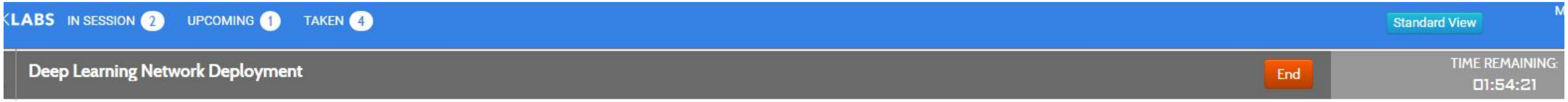
Deep Learning Network Deployment

Launching

TIME REMAINING:
01:55:00

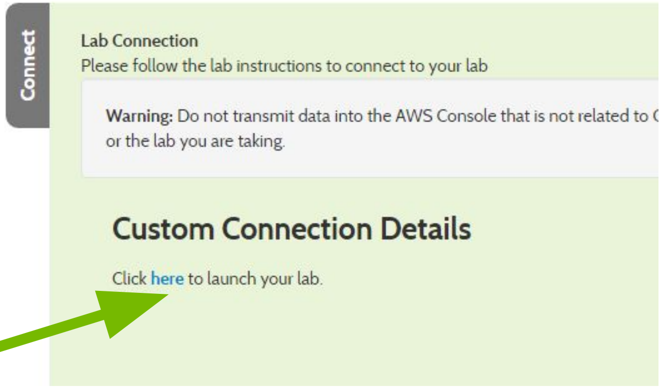
You should see that the lab environment is “launching” towards the upper-right corner

CONNECTING TO THE LAB ENVIRONMENT



Deep Learning Network Deployment

TIME REMAINING: 01:54:21



Lab Connection

Please follow the lab instructions to connect to your lab

Warning: Do not transmit data into the AWS Console that is not related to
or the lab you are taking.

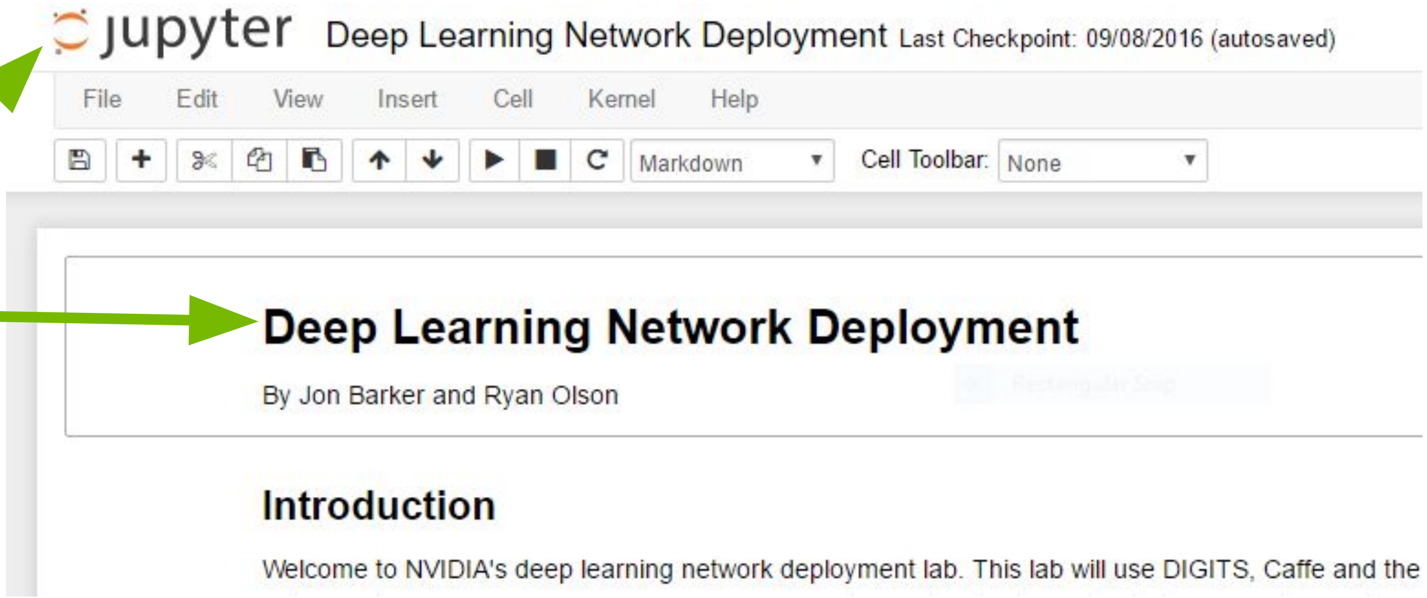
Custom Connection Details

Click here to launch your lab.

7. Click on “here” to access your lab environment / Jupyter notebook

CONNECTING TO THE LAB ENVIRONMENT

You should see your "Deep Learning Network Deployment" Jupyter notebook



Deep Learning Network Deployment Last Checkpoint: 09/08/2016 (autosaved)

Deep Learning Network Deployment

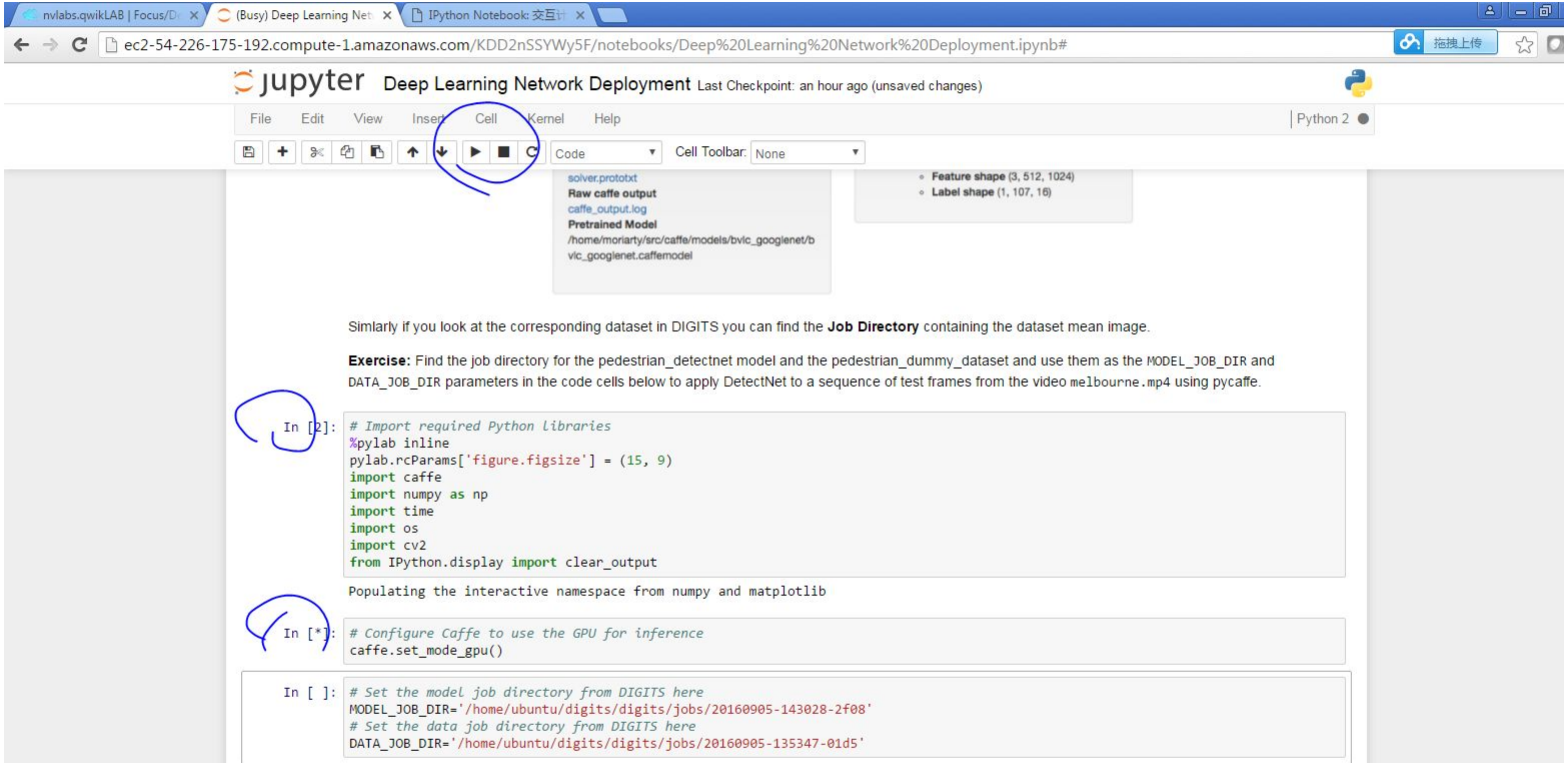
By Jon Barker and Ryan Olson

Introduction

Welcome to NVIDIA's deep learning network deployment lab. This lab will use DIGITS, Caffe and the

Jupyter Notebook Introduction

Interface: Run



Deep Learning Network Deployment Last Checkpoint: an hour ago (unsaved changes)

solver.prototxt
Raw caffe output
caffe_output.log
Pretrained Model
/home/moriarty/src/caffe/models/bvlc_googlenet/bvlc_googlenet.caffemodel

- **Feature shape** (3, 512, 1024)
- **Label shape** (1, 107, 16)

Simlarly if you look at the corresponding dataset in DIGITS you can find the **Job Directory** containing the dataset mean image.

Exercise: Find the job directory for the pedestrian_detectnet model and the pedestrian_dummy_dataset and use them as the `MODEL_JOB_DIR` and `DATA_JOB_DIR` parameters in the code cells below to apply DetectNet to a sequence of test frames from the video `melbourne.mp4` using pycaffe.

In [2]:
```
# Import required Python libraries
%pylab inline
pylab.rcParams['figure.figsize'] = (15, 9)
import caffe
import numpy as np
import time
import os
import cv2
from IPython.display import clear_output
```

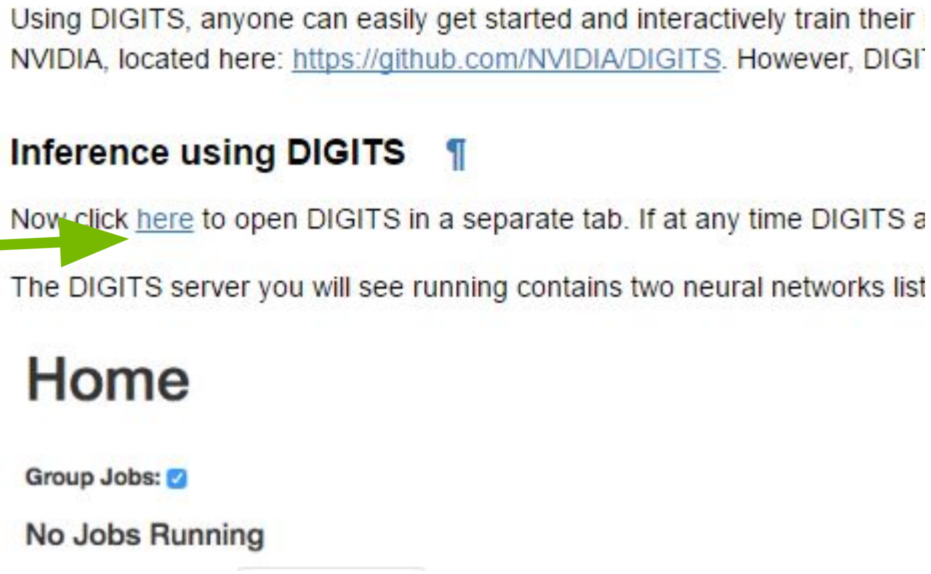
```
Populating the interactive namespace from numpy and matplotlib
```

In [*]:
```
# Configure Caffe to use the GPU for inference
caffe.set_mode_gpu()
```

In [ ]:
```
# Set the model job directory from DIGITS here
MODEL_JOB_DIR='/home/ubuntu/digits/digits/jobs/20160905-143028-2f08'
# Set the data job directory from DIGITS here
DATA_JOB_DIR='/home/ubuntu/digits/digits/jobs/20160905-135347-01d5'
```

STARTING DIGITS

Instruction in Jupyter notebook will link you to DIGITS



Using DIGITS, anyone can easily get started and interactively train their

NVIDIA, located here: https://github.com/NVIDIA/DIGITS. However, DIGI

Inference using DIGITS ¶

Now click here to open DIGITS in a separate tab. If at any time DIGITS a

The DIGITS server you will see running contains two neural networks list

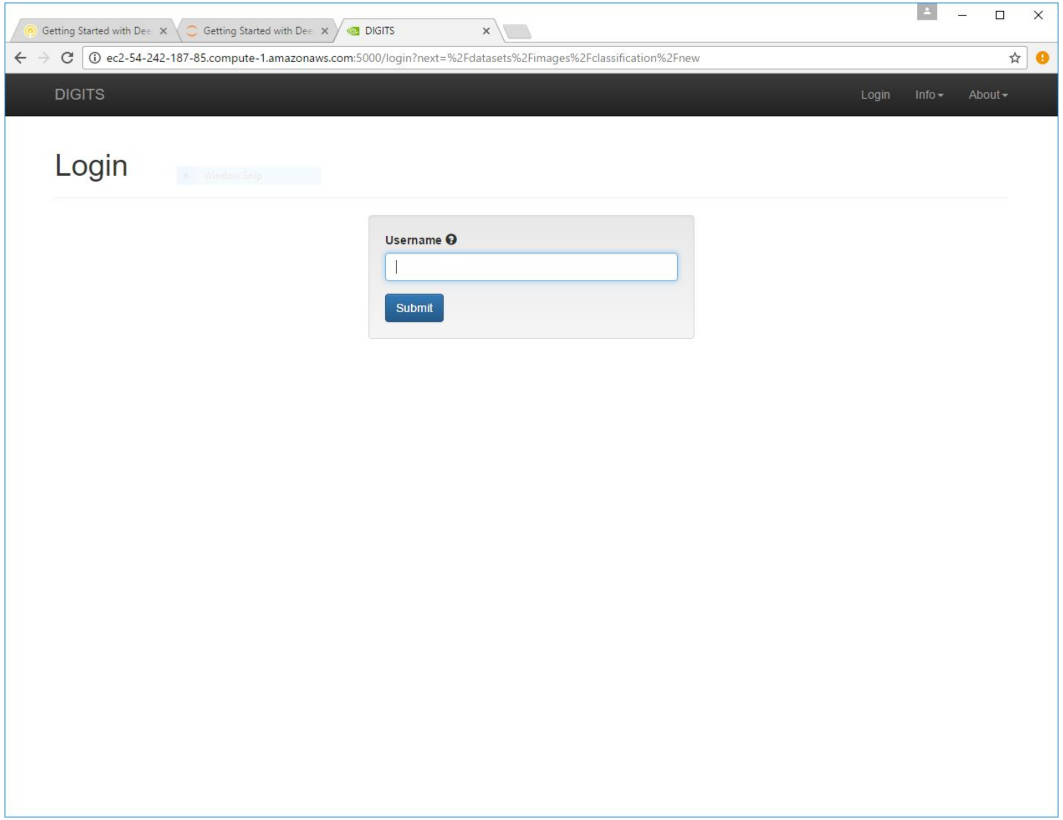
Home

Group Jobs: ☑

No Jobs Running

ACCESSING DIGITS

- Will be prompted to enter a username to access DIGITS
  - Can enter any username
  - Use lower case letters



ec2-54-242-187-85.compute-1.amazonaws.com:5000/login?next=%2Fdatasets%2Fimages%2Fclassification%2Fnew

Login

Username

REVIEW / NEXT STEPS

WHAT'S NEXT

- Use / practice what you learned
- Discuss with peers practical applications of DNN
- Reach out to NVIDIA and the Deep Learning Institute
- Look for local meetups
- Follow people like Andrej Karpathy and Andrew Ng

WHAT'S NEXT

TAKE SURVEY

…for the chance to win an NVIDIA SHIELD TV.

Check your email for a link.

ACCESS ONLINE LABS

Check your email for details to access more DLI training online.

ATTEND WORKSHOP

Visit www.nvidia.com/dli for workshops in your area.

JOIN DEVELOPER PROGRAM

Visit https://developer.nvidia.com/join for more.

GTC AROUND THE WORLD

GTC CHINA

BEIJING

SEPTEMBER 25 -27, 2017

GTC EUROPE

MUNICH

OCTOBER 10 - 12, 2017

GTC ISRAEL

TEL AVIV

OCTOBER 18, 2017

GTC DC

WASHINGTON, DC

NOVEMBER 1 - 2, 2017

GTC JAPAN

TOKYO

DECEMBER 12 - 13, 2017

GTC 2018

SILICON VALLEY

MARCH 26 - 29, 2018

WWW.GPUTECHCONF.COM

Instructor: Twin Karmakharm

DEEP
LEARNING
INSTITUTE

www.nvidia.com/dli

GPU TECHNOLOGY CONFERENCE

Join the Conversation
#GTC18

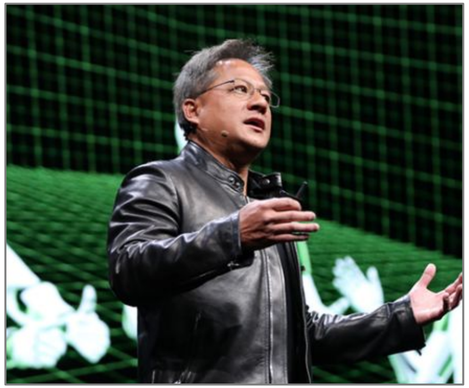
GPU TECHNOLOGY CONFERENCE

CONNECT

Connect with technology experts from NVIDIA and other leading organisations.

LEARN

Gain insight and valuable hands-on training through hundreds of sessions and research posters.



DISCOVER

Discover the latest breakthroughs in fields such as autonomous vehicles, HPC, smart cities, VR, robotics, and more.

INNOVATE

Hear about disruptive innovations as startups and researchers present their work.

USE CODE NVMDIERINGER TO SAVE 25% | REGISTER AT WWW.GPUTECHCONF.EU

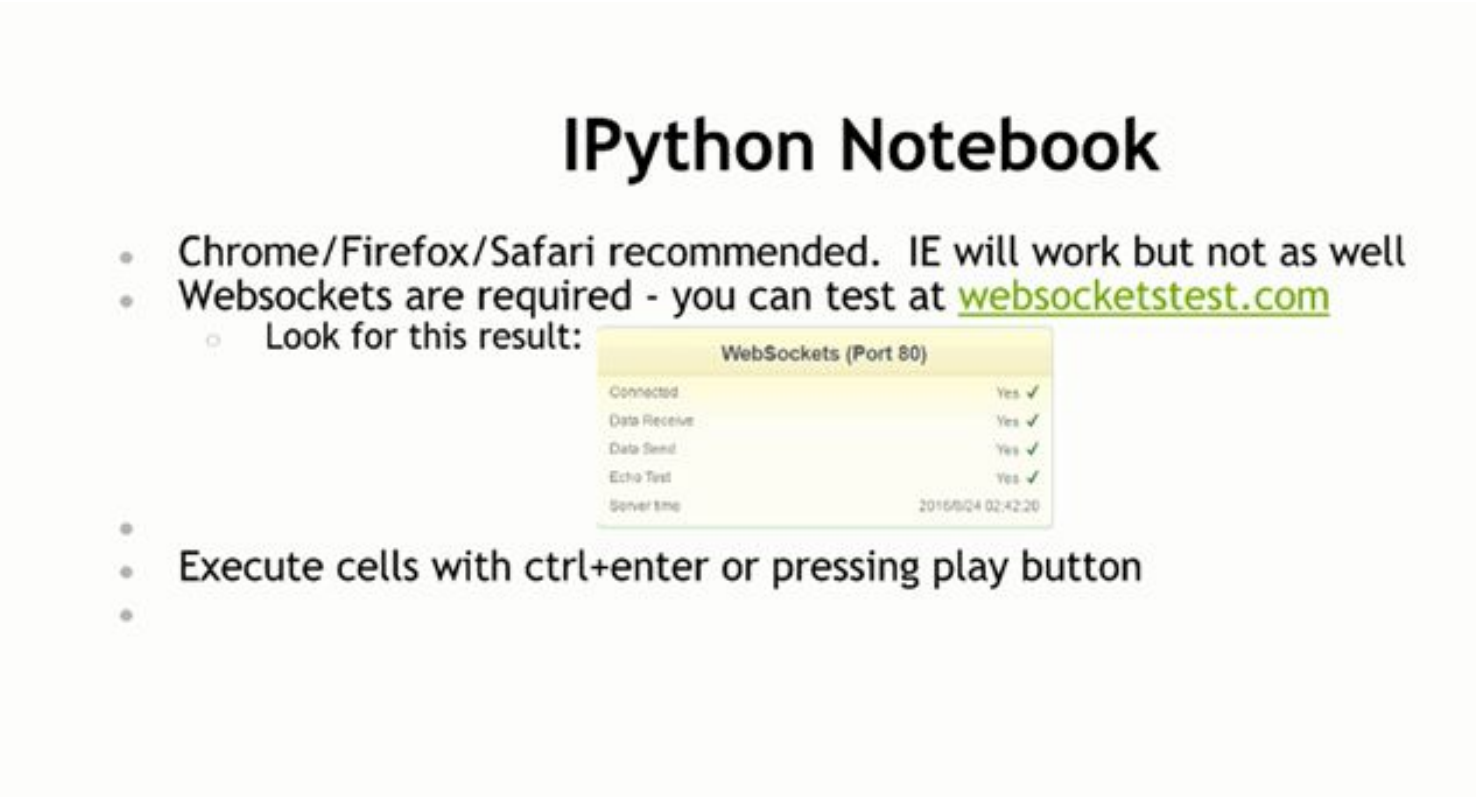
Join us at Europe’s premier conference on artificial intelligence.

9-11 October 2018 at the International Congress Centre, Munich.

APPENDIX

Lab Debug

Can’t display Ipython Notebook?



IPython Notebook

- Chrome/Firefox/Safari recommended. IE will work but not as well
- Websockets are required - you can test at websocketstest.com
  - Look for this result:

WebSockets (Port 80)	
Connected	Yes ✓
Data Receive	Yes ✓
Data Send	Yes ✓
Echo Test	Yes ✓
Server time	2016/8/24 02:42:20

- Execute cells with ctrl+enter or pressing play button

Lab Debug

Don’t know if cell is running??

You should see In[*] and not In[ ] or In[<some number>].

Solid grey circle in the top-right of the browser window

If you only see #1 and not #2, then you need to try the following in order:

Press the stop button on the toolbar. Try again.

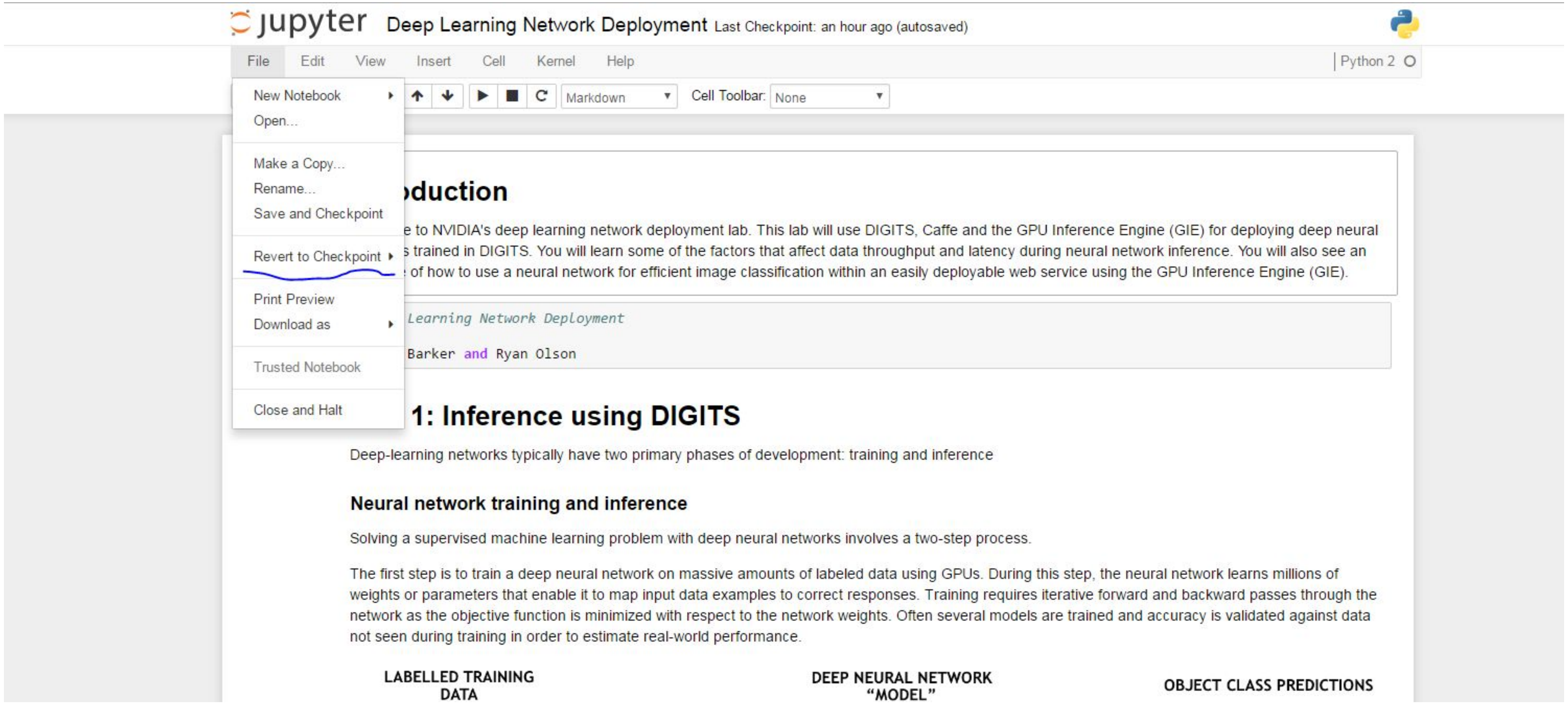
Click Kernel -> Restart. Try again.

Save the Notebook and refresh the page. Try again.

End the lab from the qwikLABS page and start a new instance. All work will be lost.
(Please let me know before you do this)

Lab Debug

Reverse to some checkpoint



›duction

e to NVIDIA's deep learning network deployment lab. This lab will use DIGITS, Caffe and the GPU Inference Engine (GIE) for deploying deep neural s trained in DIGITS. You will learn some of the factors that affect data throughput and latency during neural network inference. You will also see an : of how to use a neural network for efficient image classification within an easily deployable web service using the GPU Inference Engine (GIE).

```
Learning Network Deployment

Barker and Ryan Olson
```

1: Inference using DIGITS

Deep-learning networks typically have two primary phases of development: training and inference

Neural network training and inference

Solving a supervised machine learning problem with deep neural networks involves a two-step process.

The first step is to train a deep neural network on massive amounts of labeled data using GPUs. During this step, the neural network learns millions of weights or parameters that enable it to map input data examples to correct responses. Training requires iterative forward and backward passes through the network as the objective function is minimized with respect to the network weights. Often several models are trained and accuracy is validated against data not seen during training in order to estimate real-world performance.