Image Classification with DIGITS

Hyungon Ryu

Certified Instructor, NVIDIA Deep Learning Institute
NVIDIA Corporation

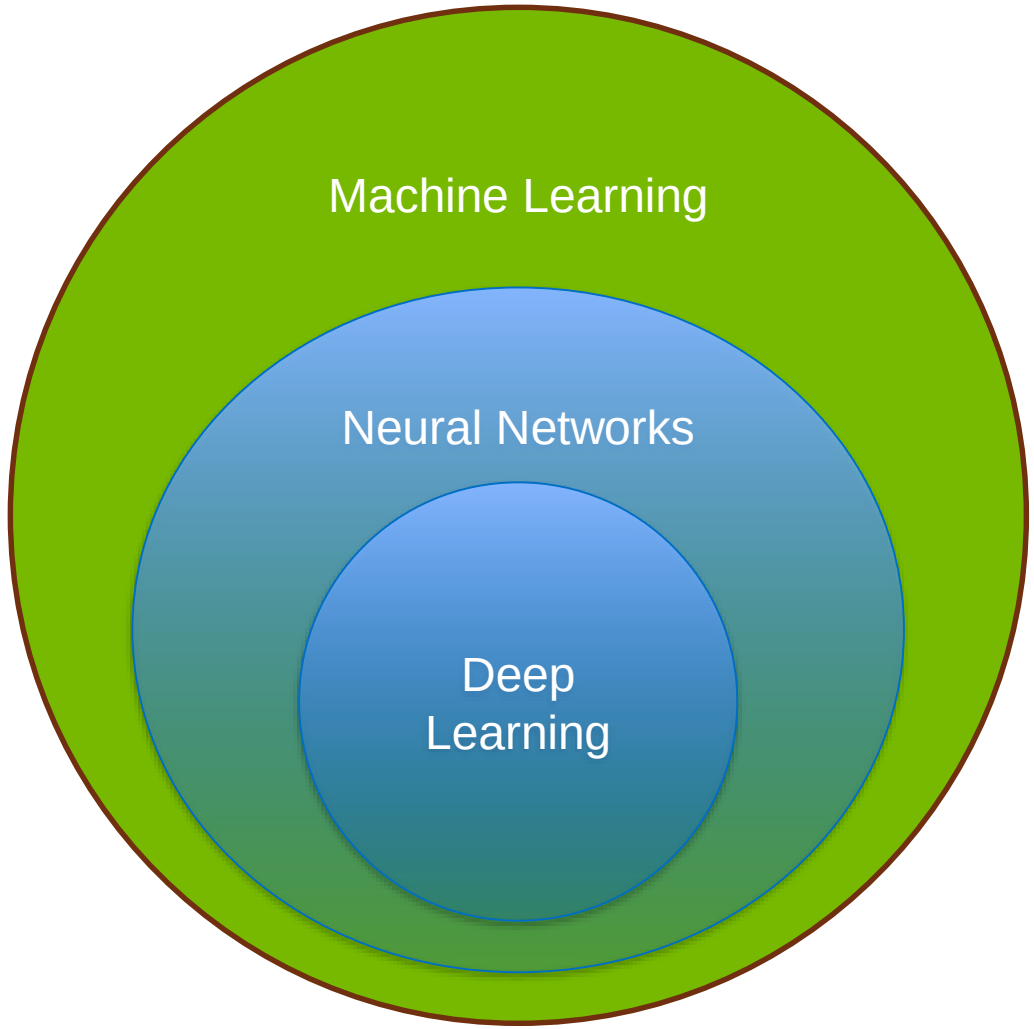
DEEP LEARNING INSTITUTE

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

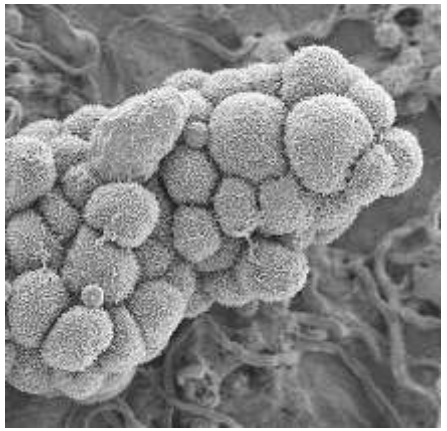
WHAT IS DEEP LEARNING?


Machine Learning
Neural Networks
Deep
Learning

DEEP LEARNING EVERYWHERE

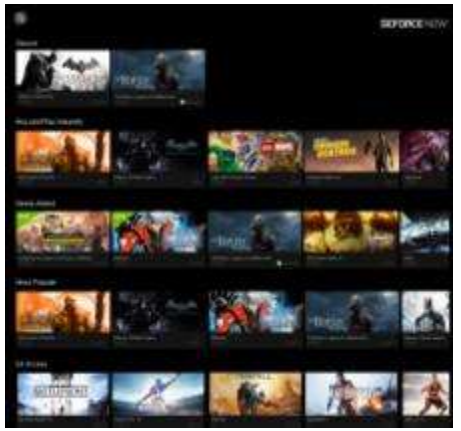
INTERNET & CLOUD

Image Classification
Speech Recognition
Language Translation
Language Processing
Sentiment Analysis
Recommendation



MEDICINE & BIOLOGY

Cancer Cell Detection
Diabetic Grading
Drug Discovery

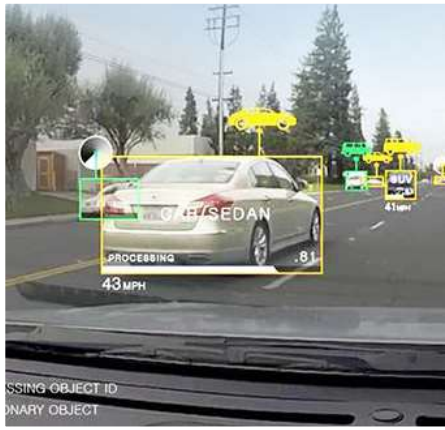


MEDIA & ENTERTAINMENT

Video Captioning
Video Search
Real Time Translation

SECURITY & DEFENSE

Face Detection
Video Surveillance
Satellite Imagery

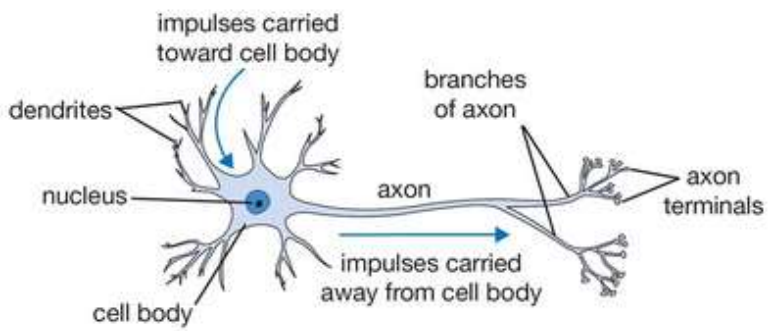


AUTONOMOUS MACHINES

Pedestrian Detection
Lane Tracking
Recognize Traffic Sign

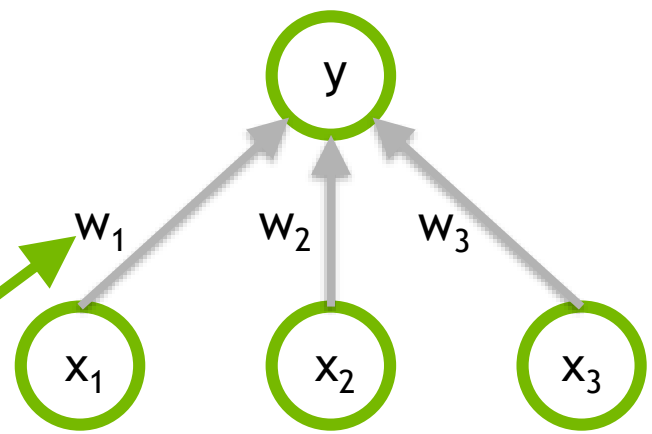
ARTIFICIAL NEURONS

Biological neuron



From Stanford cs231n lecture notes

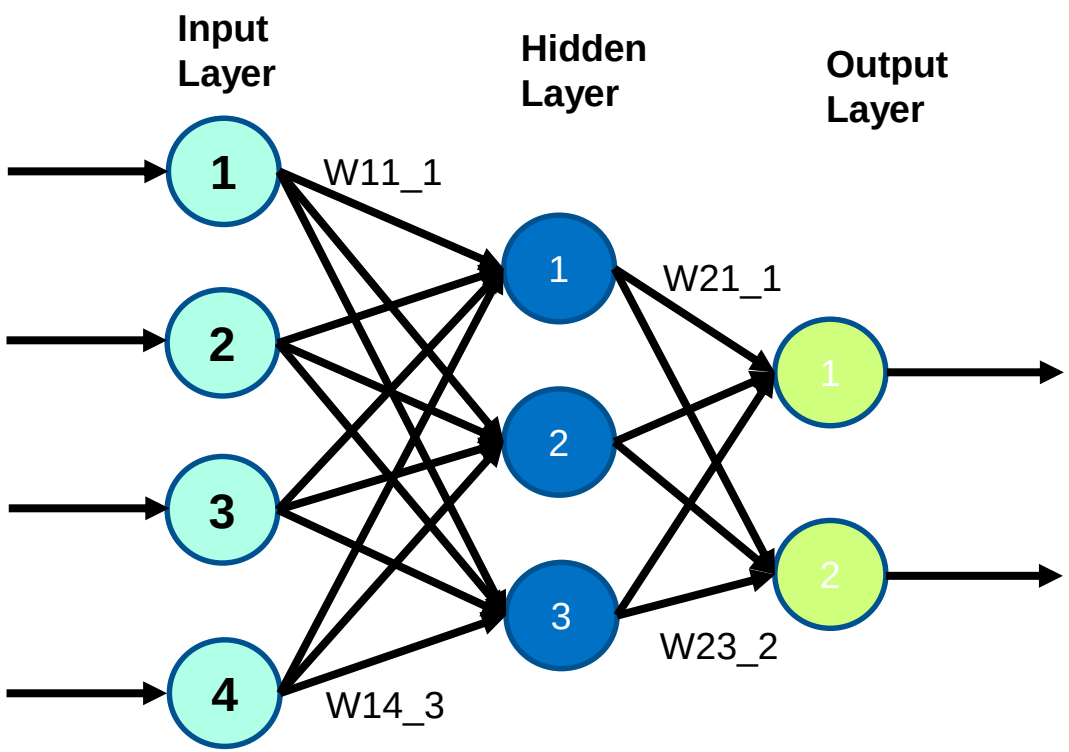
Artificial neuron



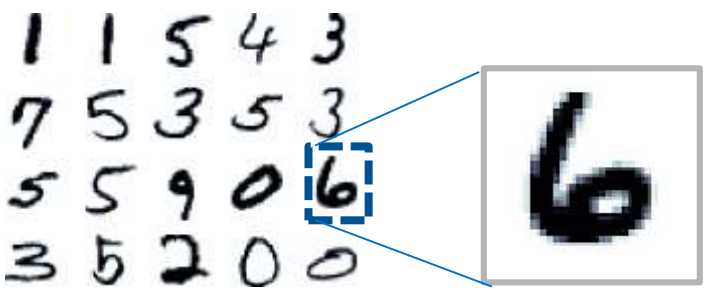
Weights (W_n)
= parameters

$y=F(w_1x_1+w_2x_2+w_3x_3)$

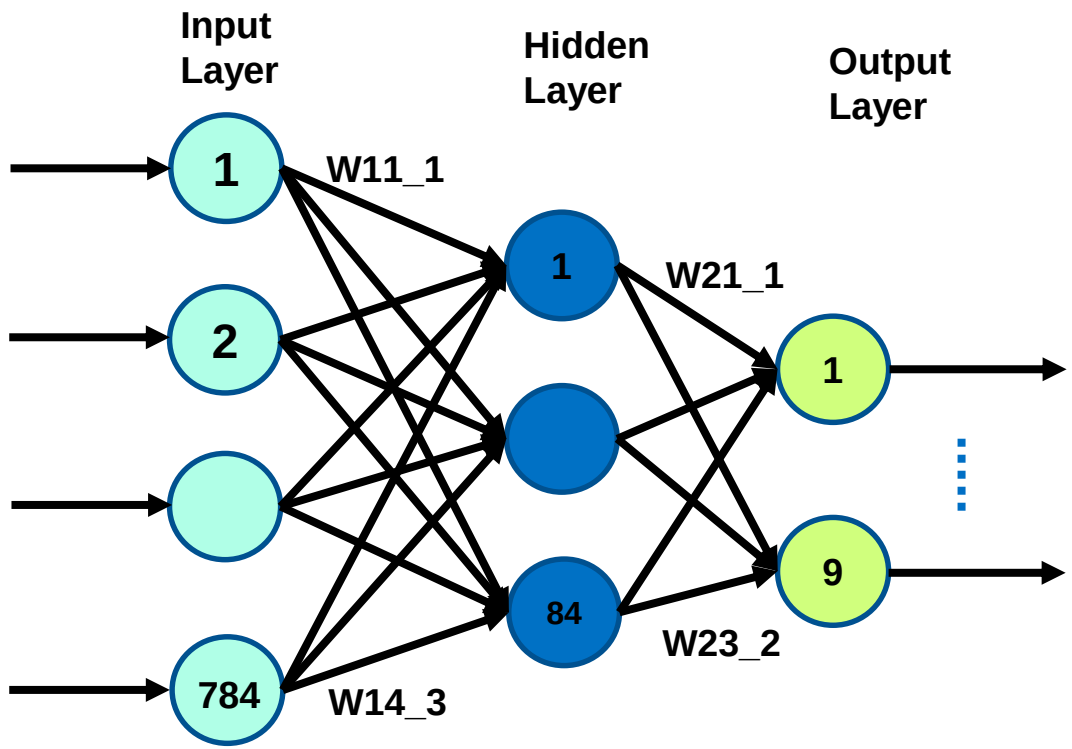
MLP


Input Layer
Hidden Layer
Output Layer
1
2
3
4
W11_1
W14_3
1
2
3
W21_1
W23_2
1
2

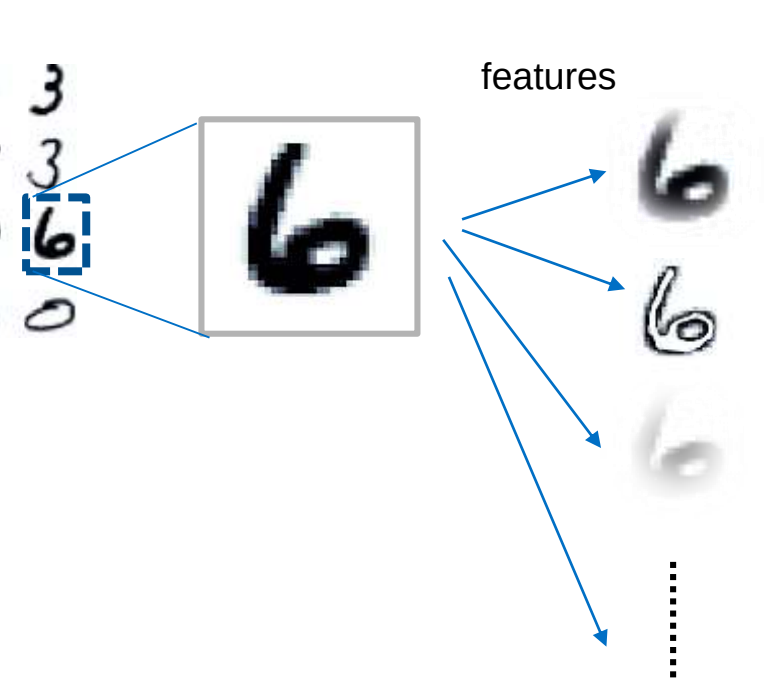
ANN for MNIST



28x28=784

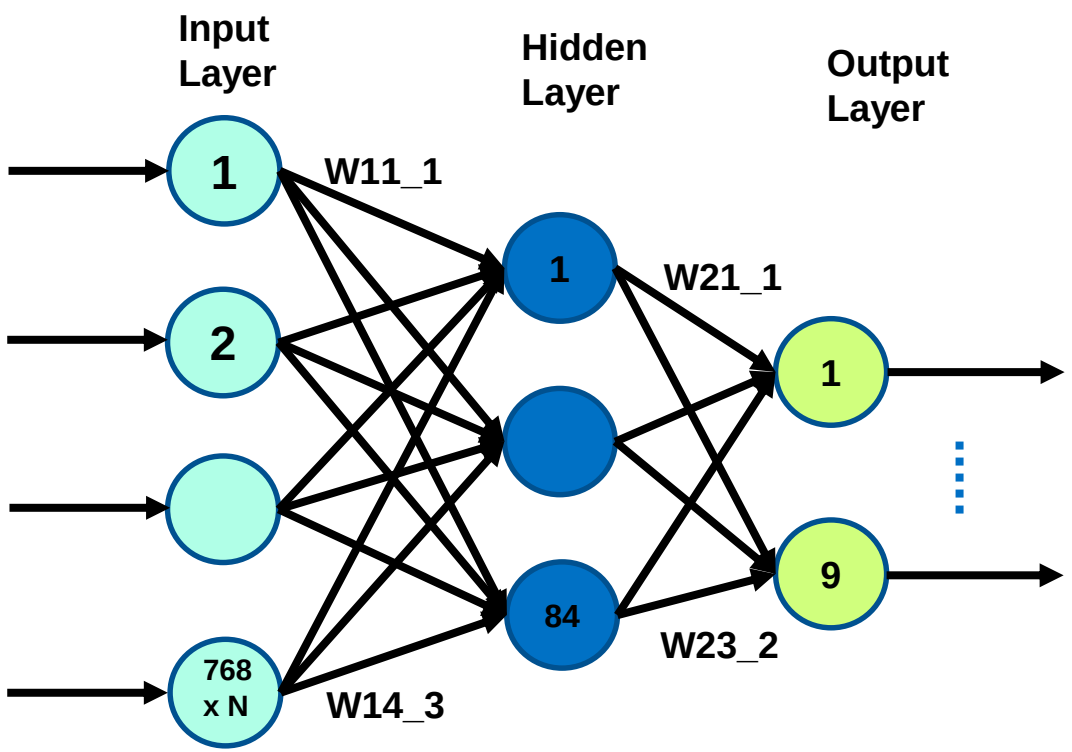


Pre-processing + ANN for MNIST



features

768 x N



Feature with Convolution Filter

edge

```
 1  2  1
 0  0  0
-1 -2 -1
```

sharpen

```
-1  -1  -1
-1   9  -1
-1  -1  -1
```

blur

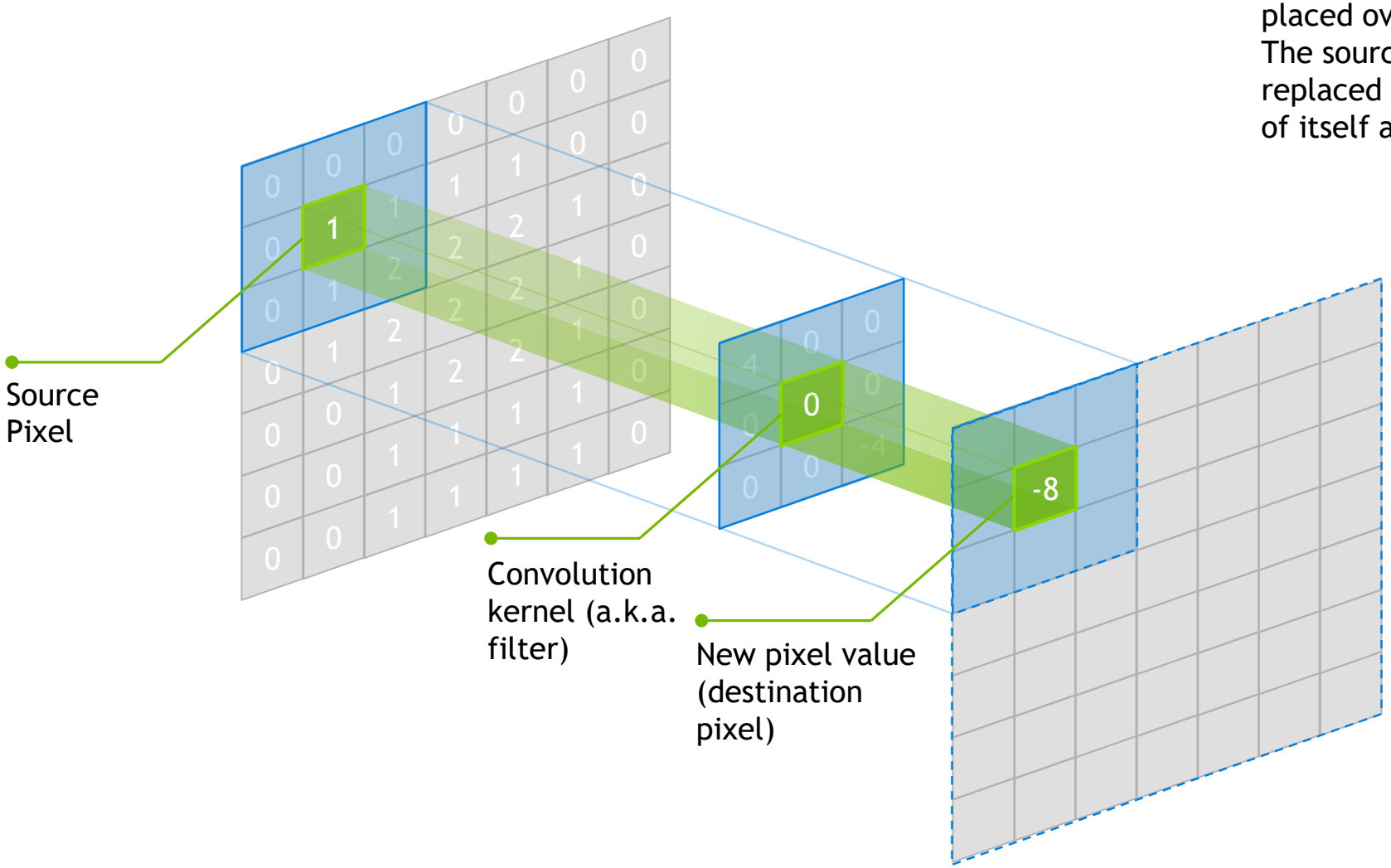
```
0.1 0.1 0.1
0.1 0.1 0.1
0.1 0.1 0.1
```

Motion blur

```
0    0    0.3
0    0.3  0
0.3  0    0
```

CONVOLUTION

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

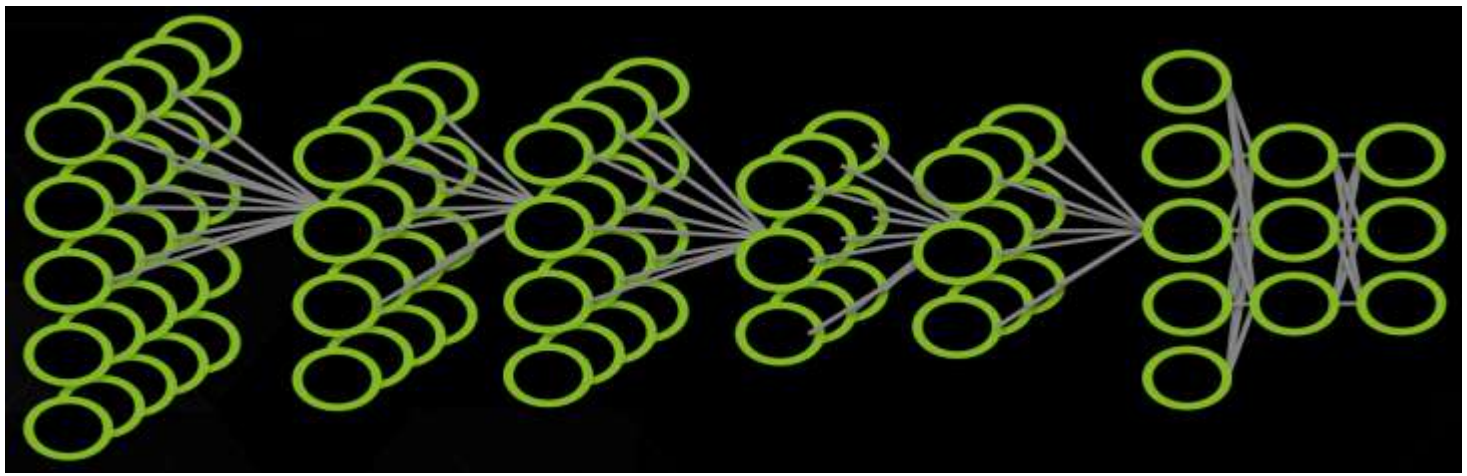


ARTIFICIAL NEURAL NETWORK

A collection of simple, trainable mathematical units that collectively learn complex functions

Hidden layers

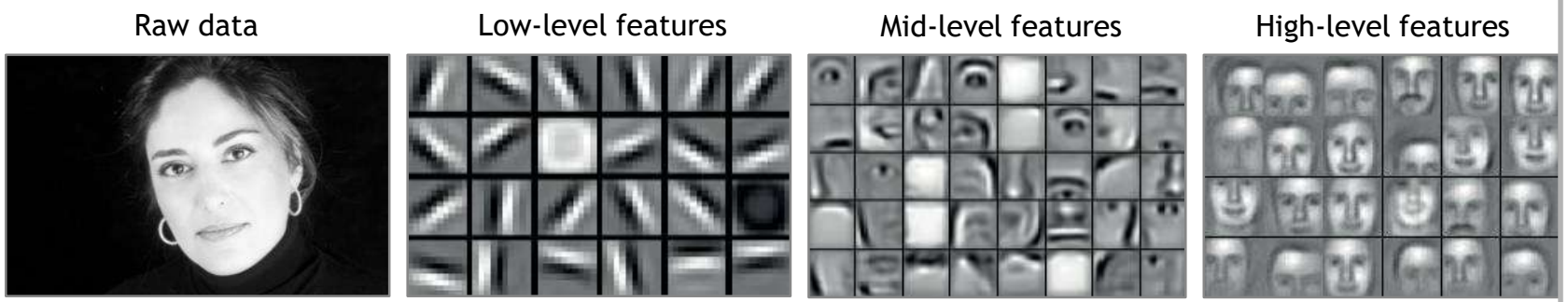
Input layer

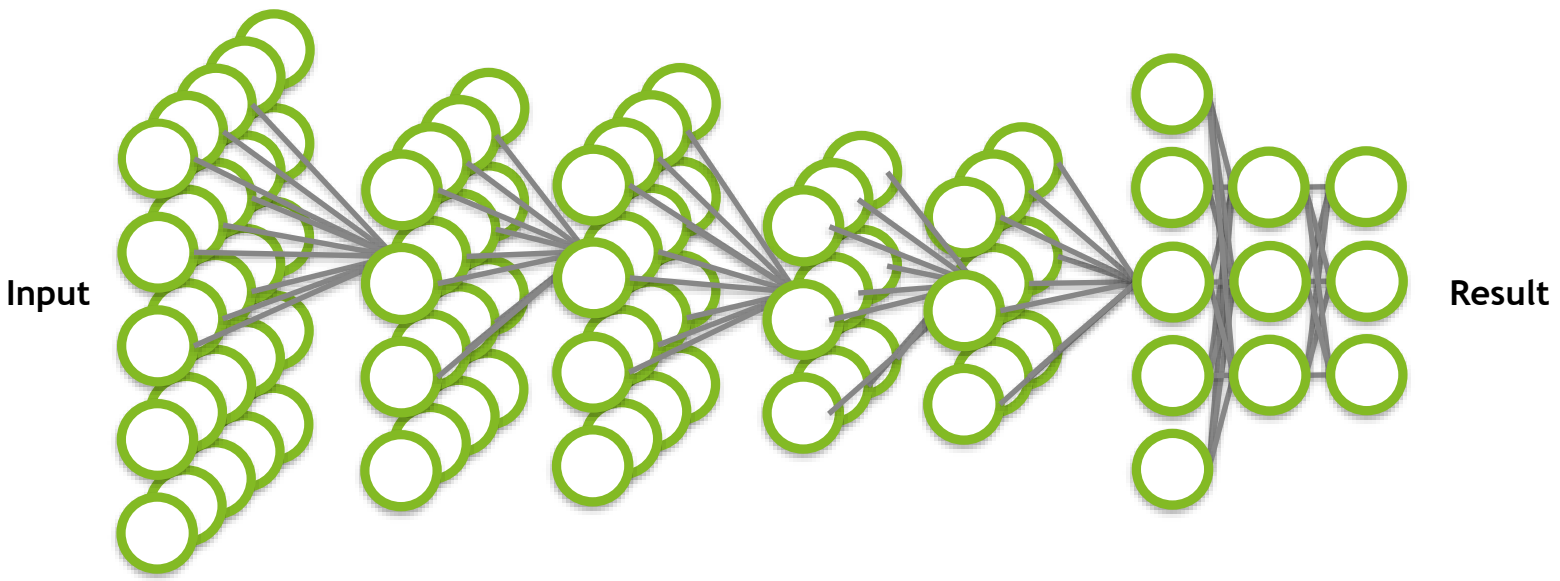


Output layer

Given sufficient training data an artificial neural network can approximate very complex functions mapping raw data to output decisions

DEEP NEURAL NETWORK (DNN)





Application components:

Task objective
e.g. Identify face

Training data
10-100M images

Network architecture
~10s-100s of layers
1B parameters

Learning algorithm
~30 Exaflops
1-30 GPU days

DEEP LEARNING APPROACH

Train:

Errors

DNN

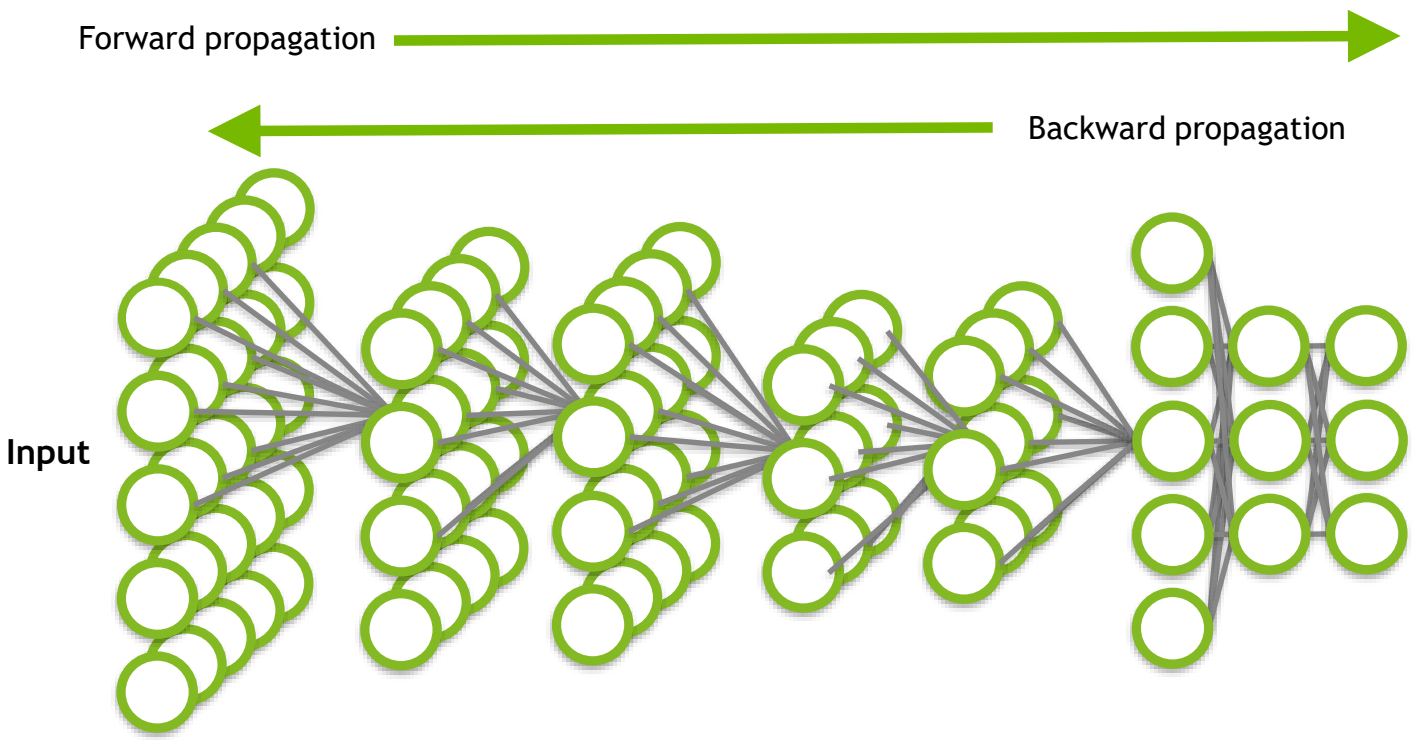
Dog
Cat
Raccoon

Deploy:

DNN

Dog

DEEP LEARNING APPROACH - TRAINING



Process

- **Forward propagation yields an inferred label for each training image**
- **Loss function used to calculate difference between known label and predicted label for each image**
- **Weights are adjusted during backward propagation**
- **Repeat the process**

ADDITIONAL TERMINOLOGY

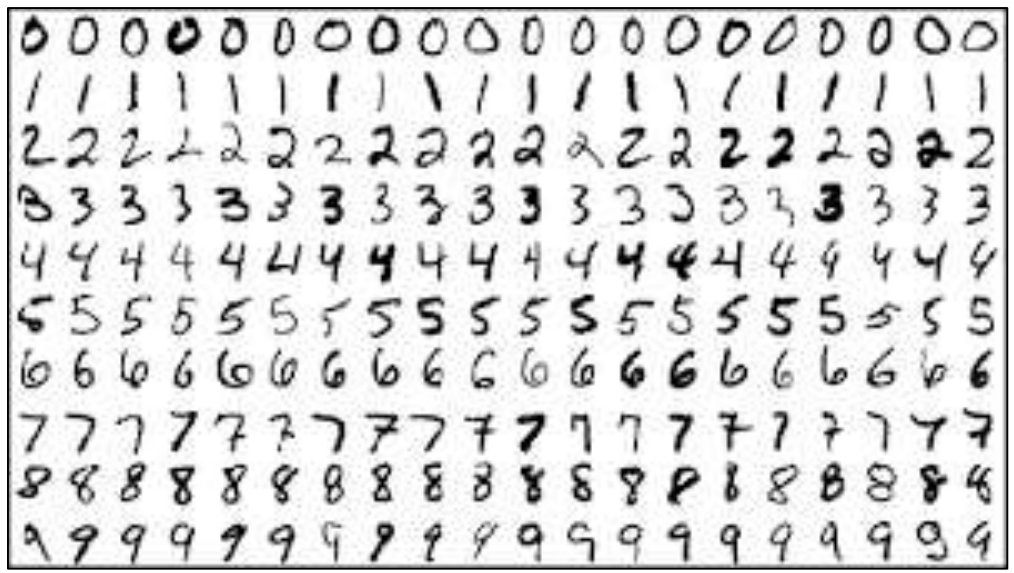
- Hyperparameters – parameters specified before training begins
  - Can influence the speed in which learning takes place
  - Can impact the accuracy of the model
  - Examples: Learning rate, decay rate, batch size
- Epoch – complete pass through the training dataset
- Activation functions – identifies active neurons
  - Examples: Sigmoid, Tanh, ReLU
- Pooling – Down-sampling technique
  - No parameters (weights) in pooling layer

HANDWRITTEN DIGIT RECOGNITION

HANDWRITTEN DIGIT RECOGNITION

HELLO WORLD of machine learning?

- MNIST data set of handwritten digits from Yann Lecun's website
- All images are 28x28 grayscale
  - Pixel values from 0 to 255
- 60K training examples / 10K test examples
- Input vector of size 784
  - 28 * 28 = 784
- Output value is integer from 0-9



CAFFE

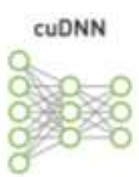
NVIDIA Powers Deep Learning

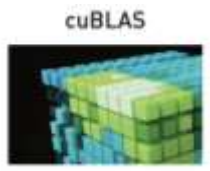
Every major DL framework leverages NVIDIA SDKs

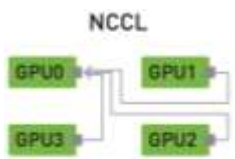
COMPUTER VISION		SPEECH & AUDIO		NATURAL LANGUAGE PROCESSING	
OBJECT DETECTION	IMAGE CLASSIFICATION	VOICE RECOGNITION	LANGUAGE TRANSLATION	RECOMMENDATION ENGINES	SENTIMENT ANALYSIS

Caffe
Chainer
DL4J
Deeplearning4j
Mocha.jl
julia
K
KERAS
Microsoft
CNTK
mxnet
Purine
TensorFlow
torch
MatConvNet
MINERVA
OpenDeep
Pylearn2
theano

NVIDIA DEEP LEARNING SDK







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"
        }
}
```

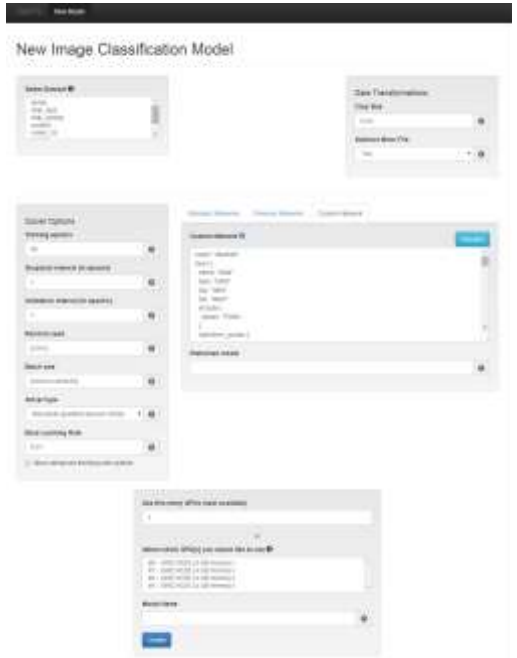
NVIDIA’S DIGITS

NVIDIA DIGITS

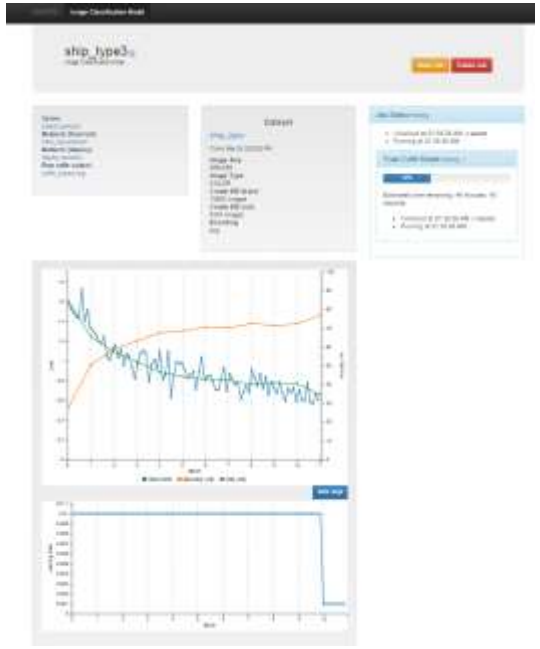
Interactive Deep Learning GPU Training System

Process Data

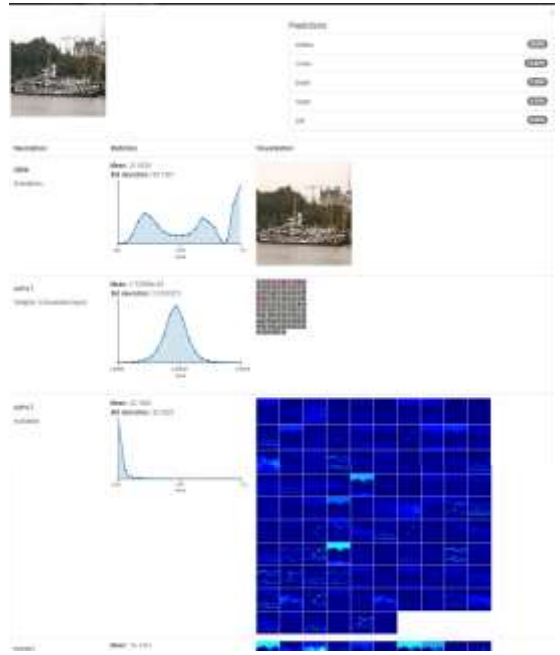
Configure DNN



Monitor Progress



Visualization



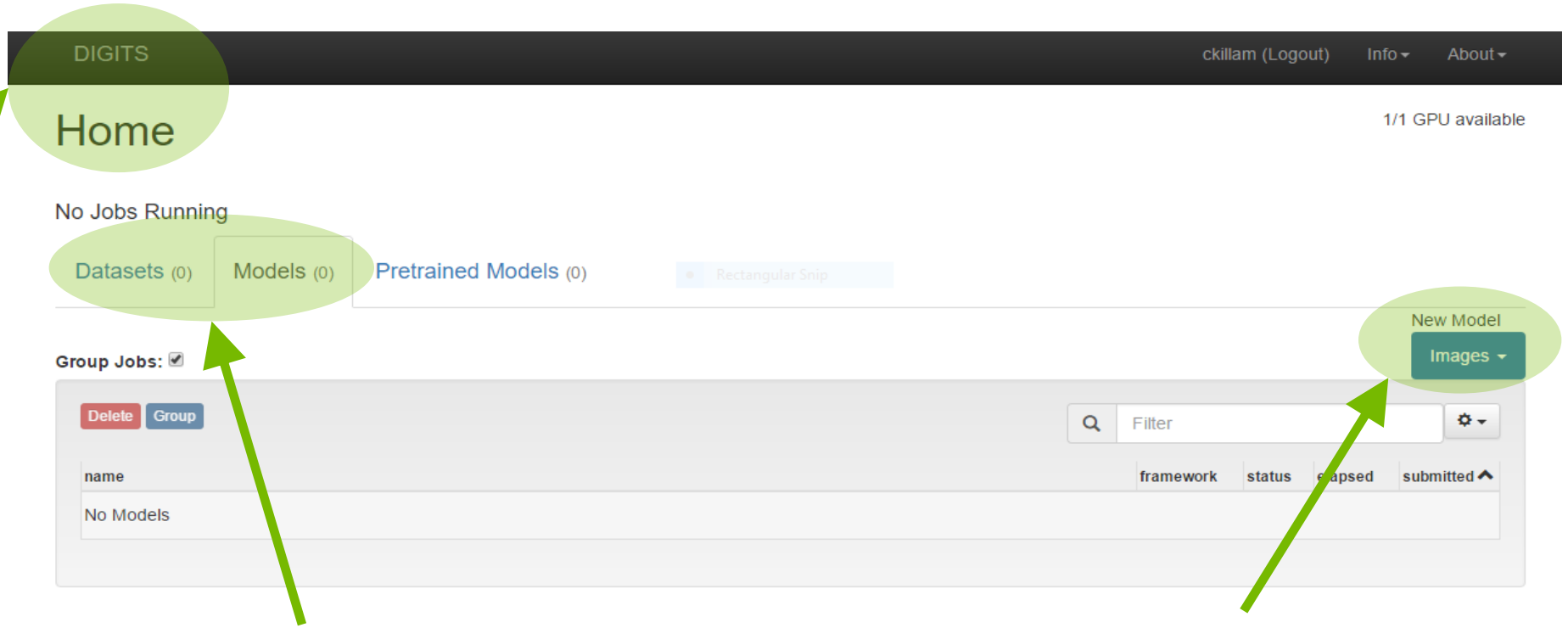
NVIDIA'S DIGITS

Interactive Deep Learning GPU Training System

- Simplifies common deep learning tasks such as:
  - Managing data
  - Designing and training neural networks on multi-GPU systems
  - Monitoring performance in real time with advanced visualizations
- Completely interactive so data scientists can focus on designing and training networks rather than programming and debugging
- Open source

DIGITS - HOME

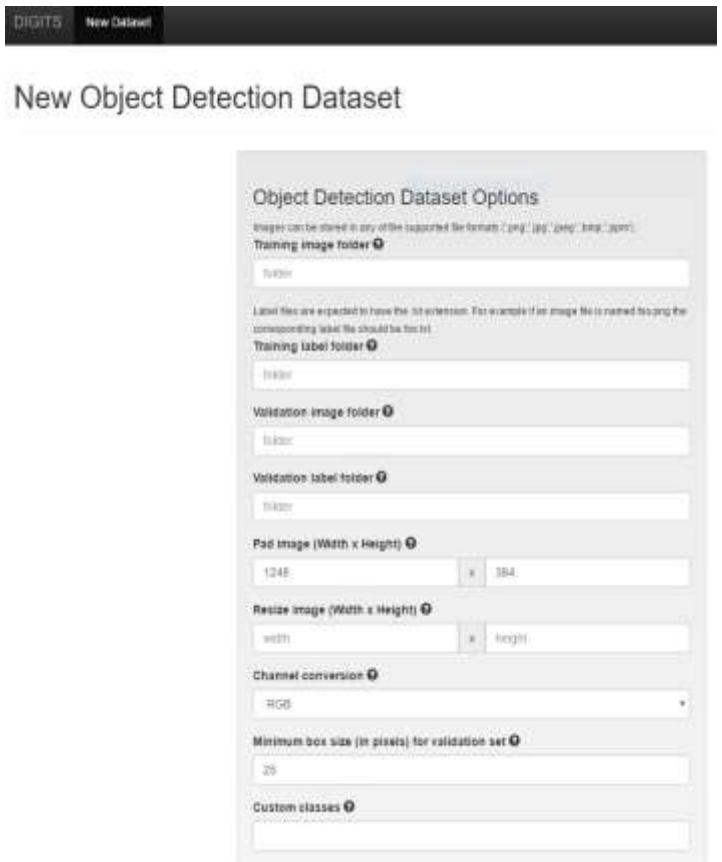
Clicking DIGITS will bring you to this Home screen

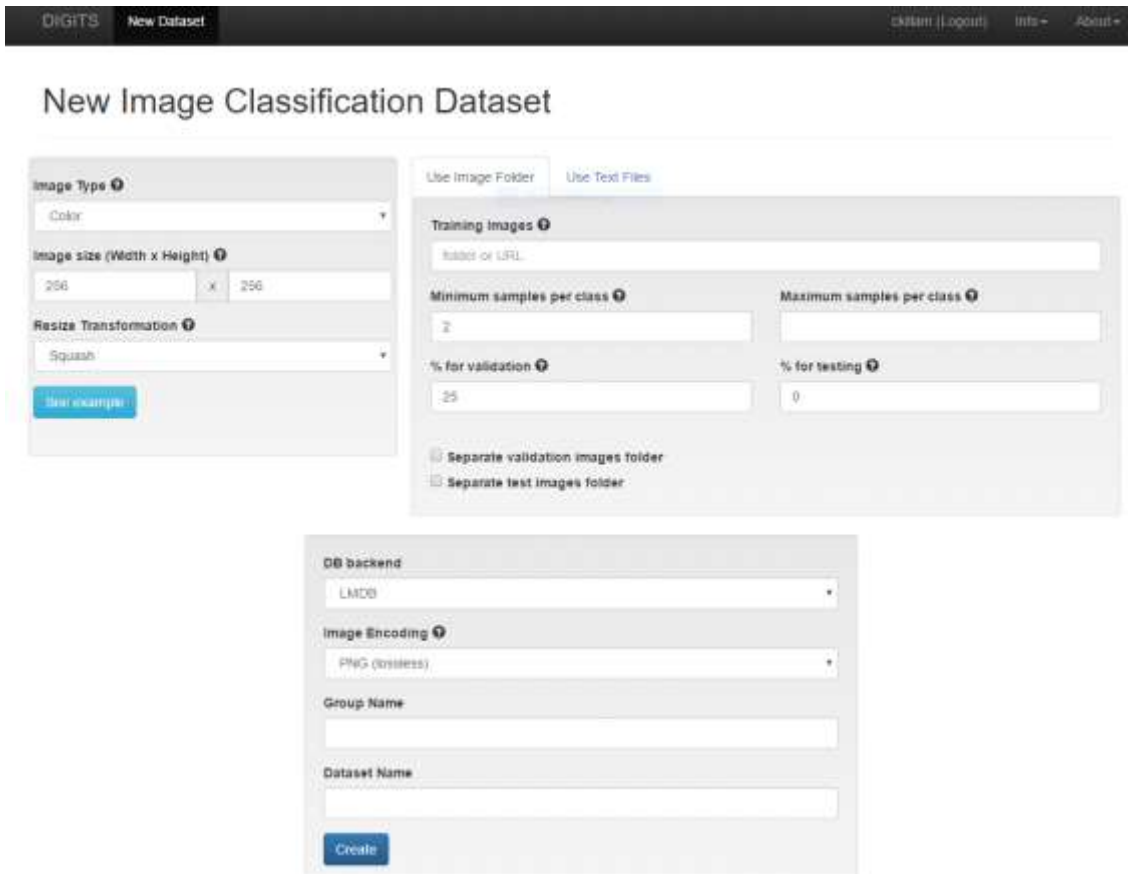


Click here to see a list of existing datasets or models

Clicking here will present different options for model and dataset creation

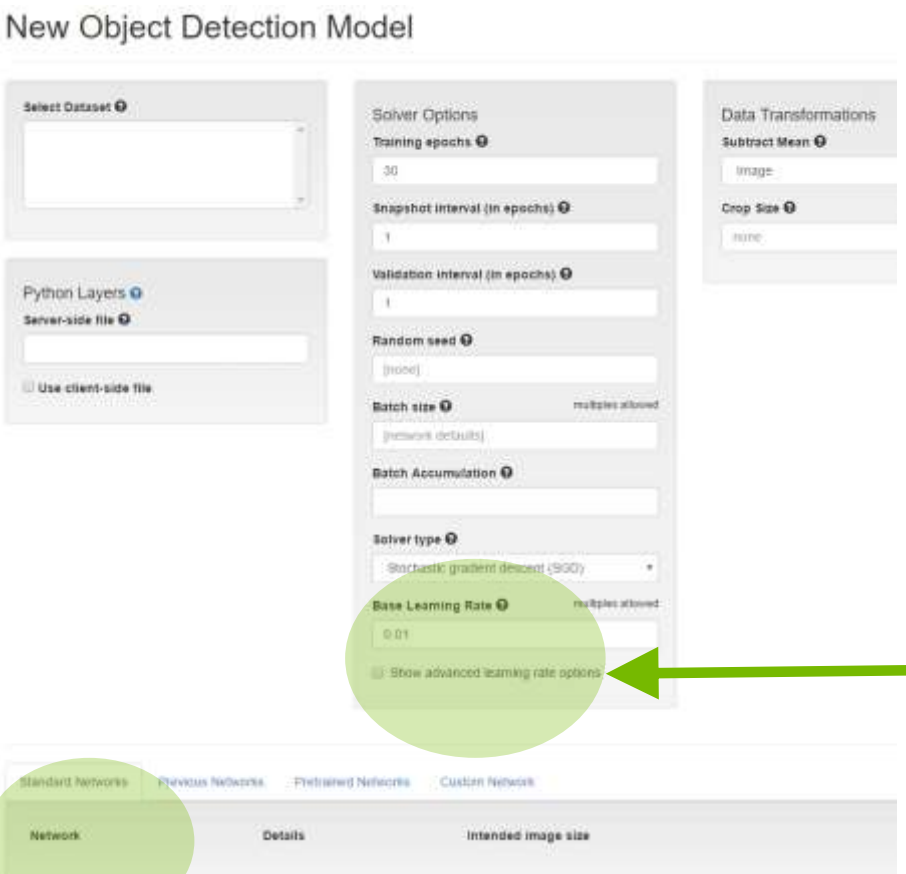
DIGITS - DATASET





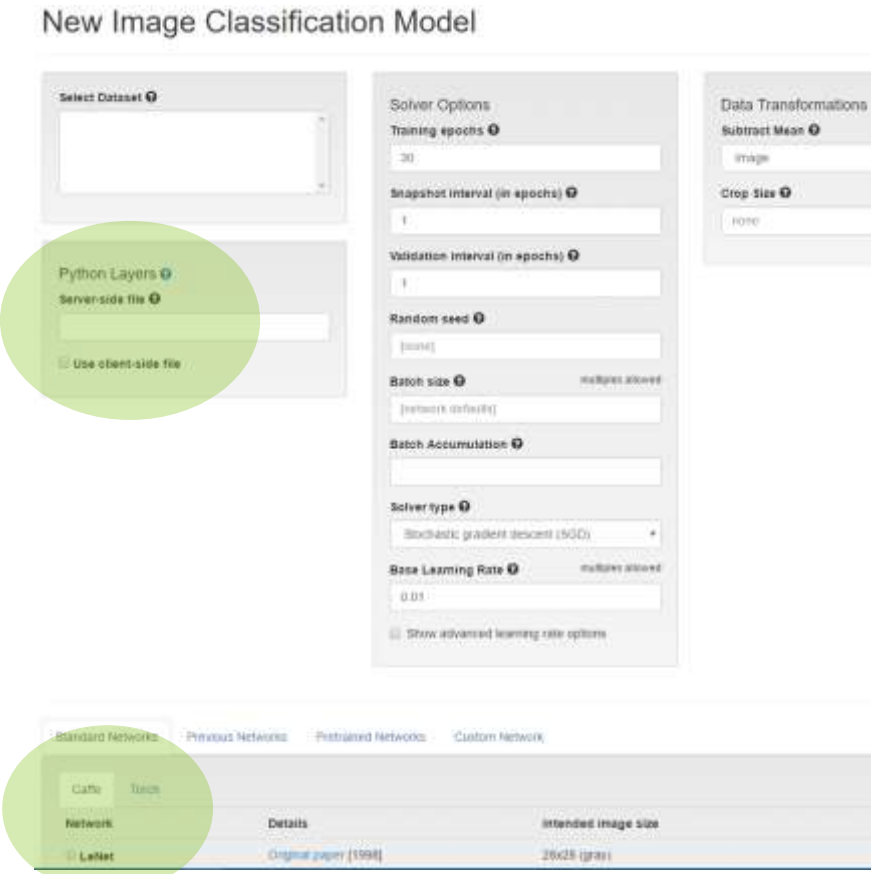
Different options will be presented based upon the task

DIGITS - MODEL



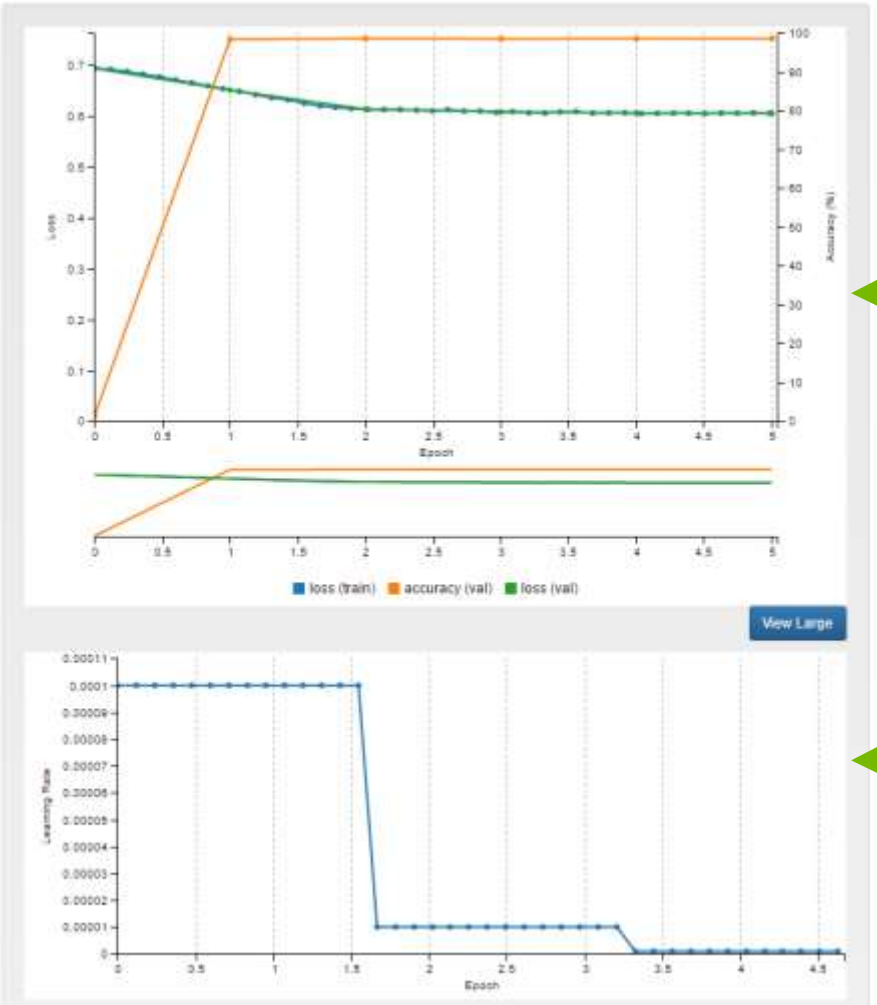
Define custom layers with Python

Can anneal the learning rate



Differences may exist between model tasks

DIGITS - TRAINING

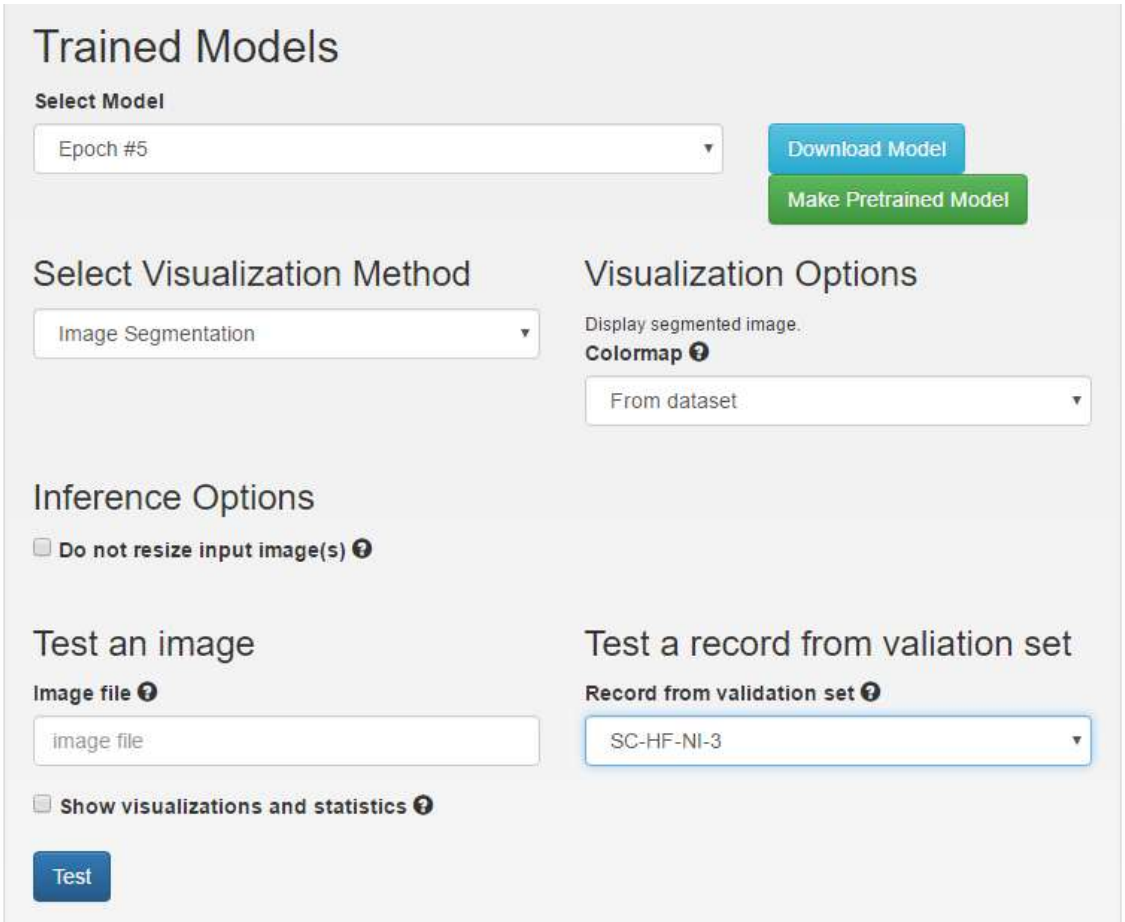


Loss function and accuracy during training

Annealed learning rate

DIGITS - VISUALIZATION

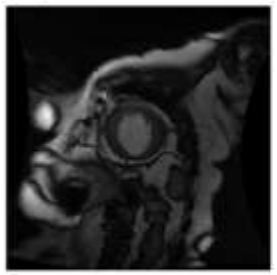
Once training is complete DIGITS provides an easy way to visualize what happened



DIGITS – VISUALIZATION RESULTS

Summary

Output visualizations



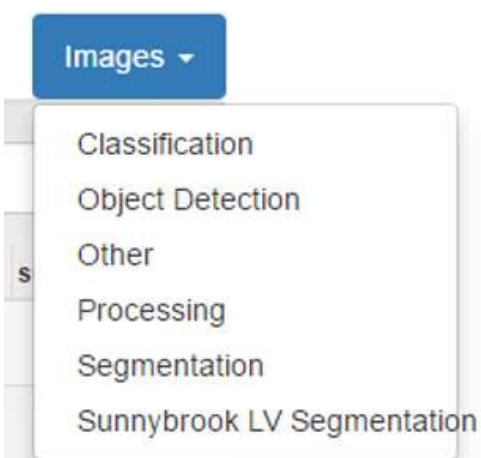
Layer visualizations

Description	Statistics	Visualization
"data" Activation	**Data shape:** [1 256 256] **Mean:** 3.27138 **Std deviation:** 75.5979	
"conv1" Weights *(Convolution layer)* 11,712 learned parameters	**Data shape:** [96 1 11 11] **Mean:** 0.0 **Std deviation:** 0.0	

DIGITS PLUGINS

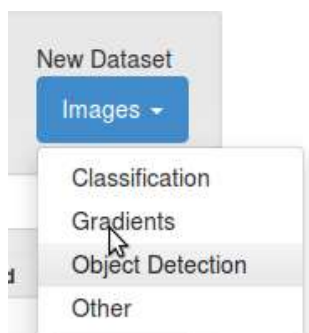
DIGITS Plugins
Image : Sunnybrook LV Segmentation

plugins/data/sunnybrook



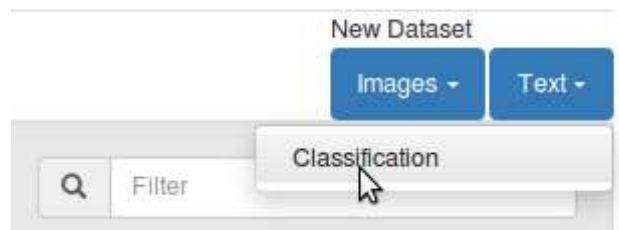
DIGITS Plugins
Image : Regression

plugins/data/imageGradients



DIGITS Plugins
Text

plugins/data/textClassification

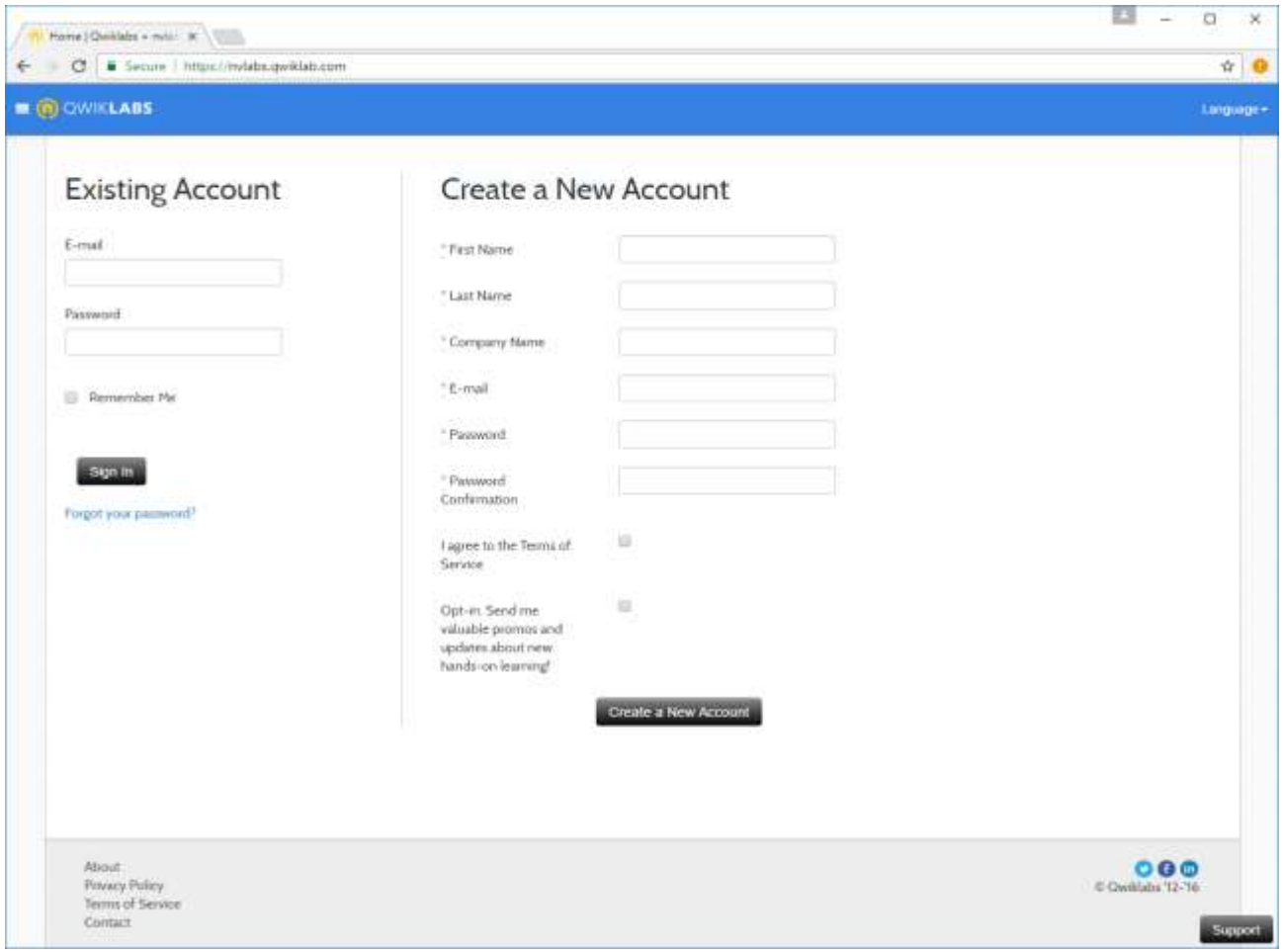


LAUNCHING THE LAB ENVIRONMENT

NAVIGATING TO QWIKLABS

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

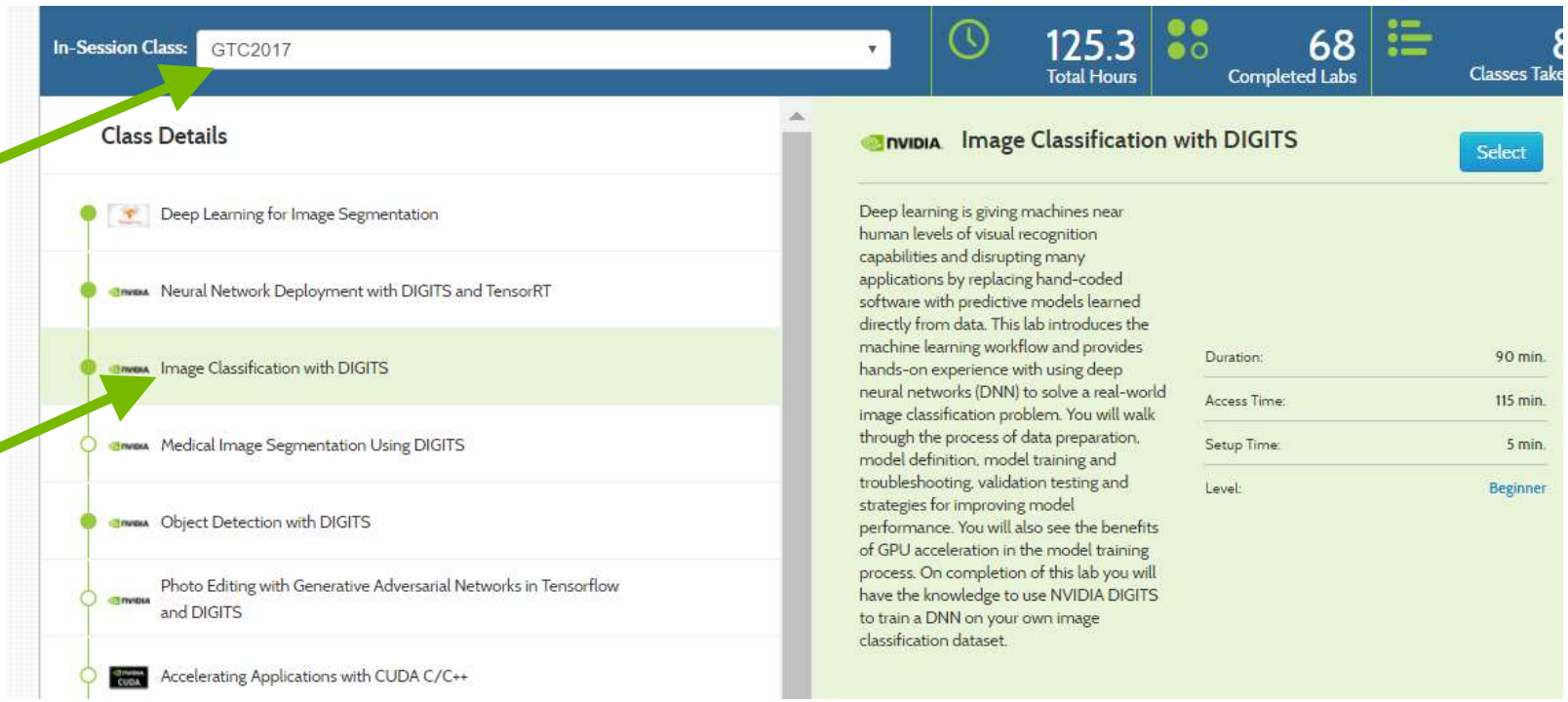
1. Login or create a new account



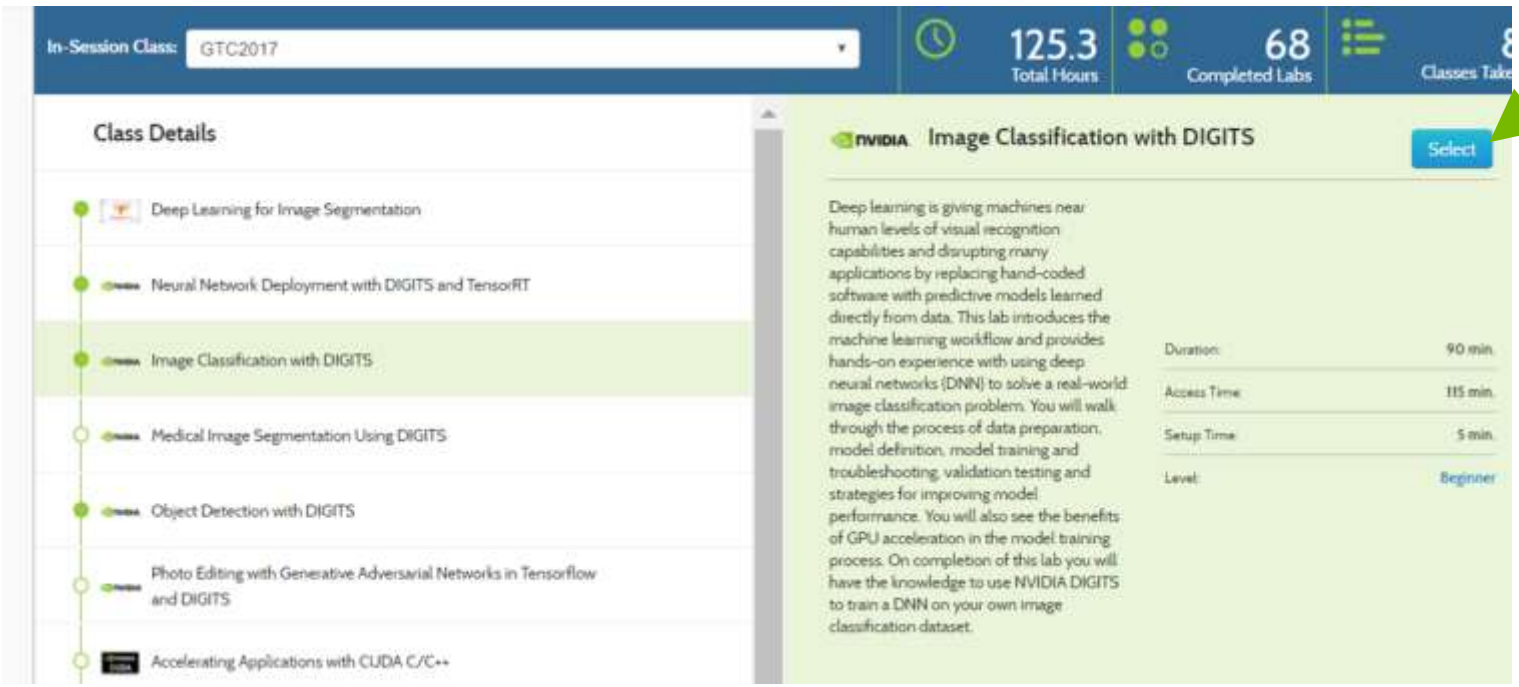
ACCESSING LAB ENVIRONMENT

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

3. Click the “Image Classification with DIGITS” Class from the list



LAUNCHING THE LAB ENVIRONMENT



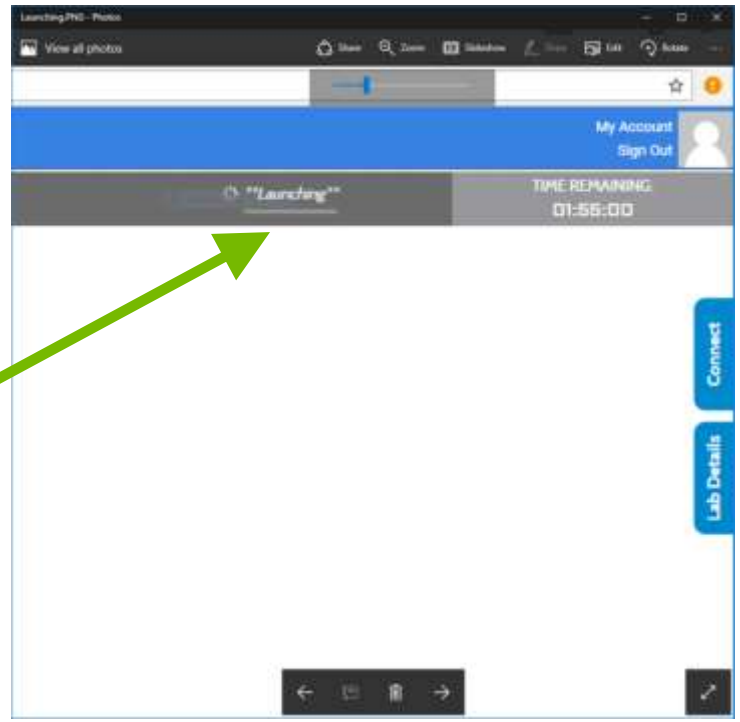
5. Click on the Select button to launch the lab environment

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

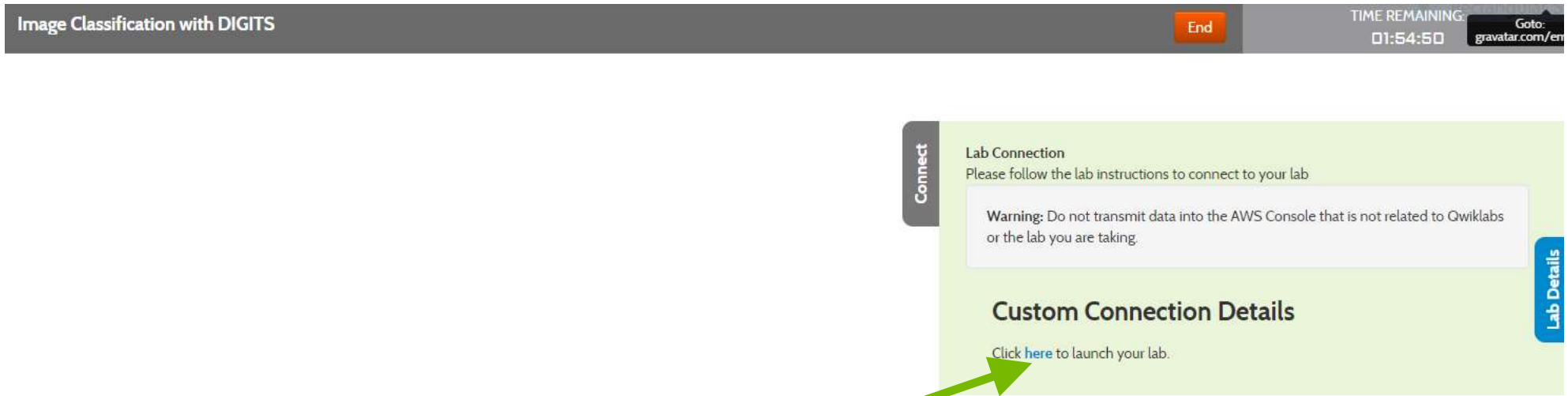
LAUNCHING THE LAB ENVIRONMENT

6. Click on the Start Lab button

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



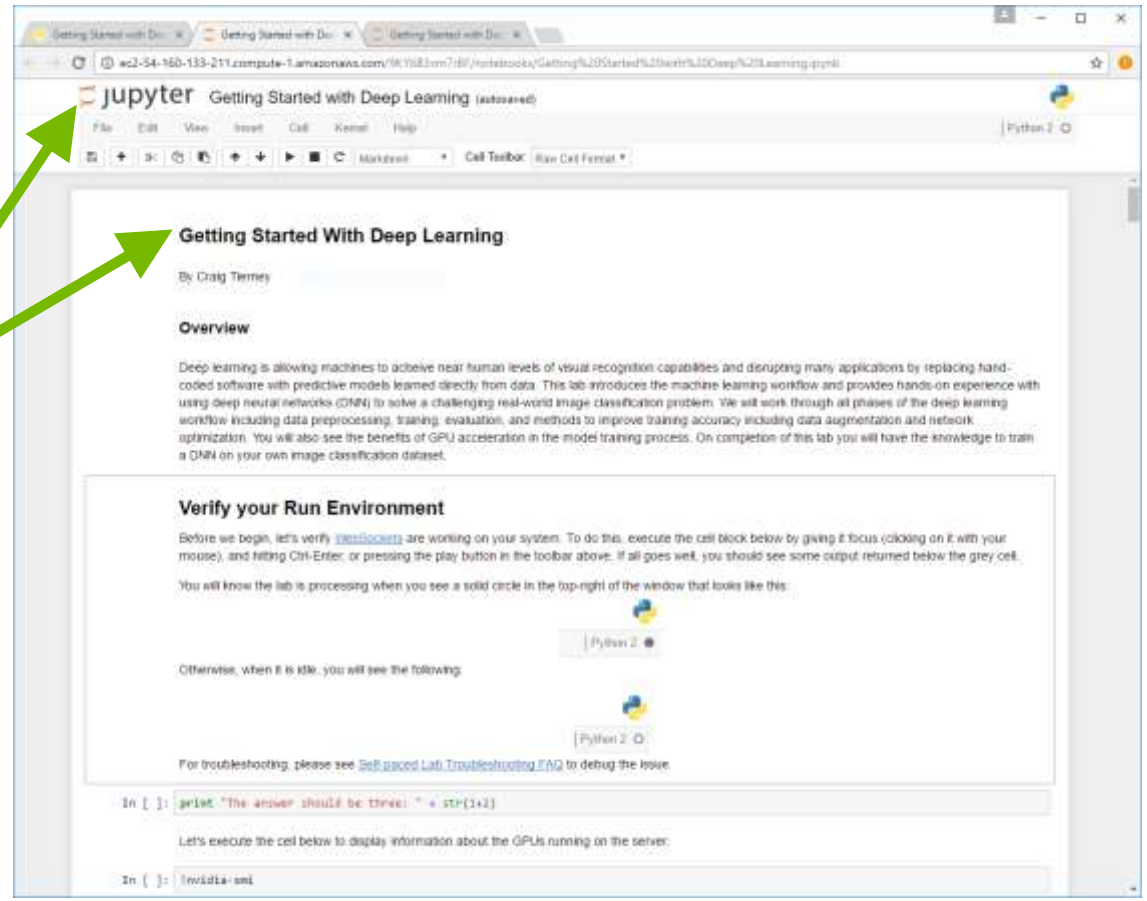
CONNECTING TO THE LAB ENVIRONMENT



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

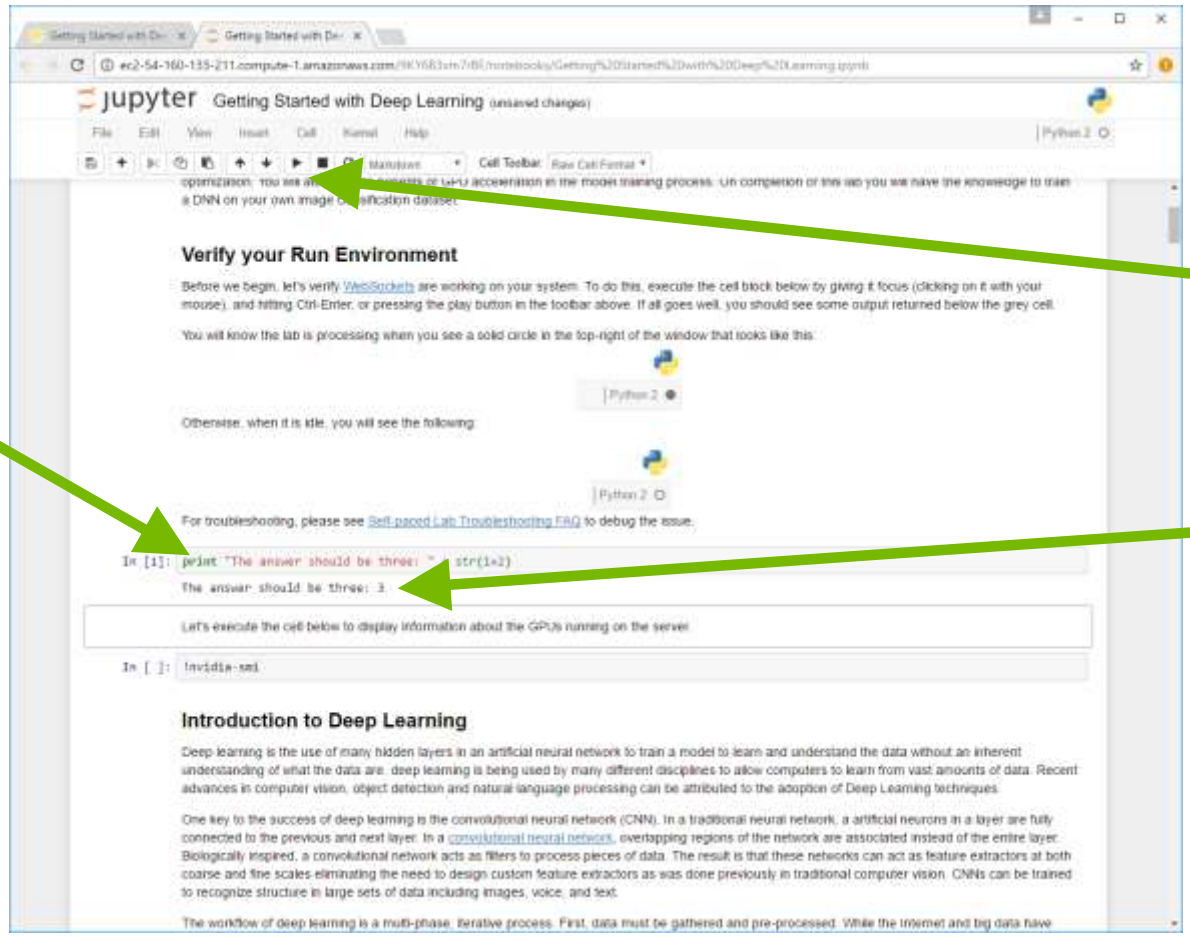
CONNECTING TO THE LAB ENVIRONMENT

You should see your “Getting Started With Deep Learning” Jupyter notebook



JUPYTER NOTEBOOK

1. Place your cursor in the code

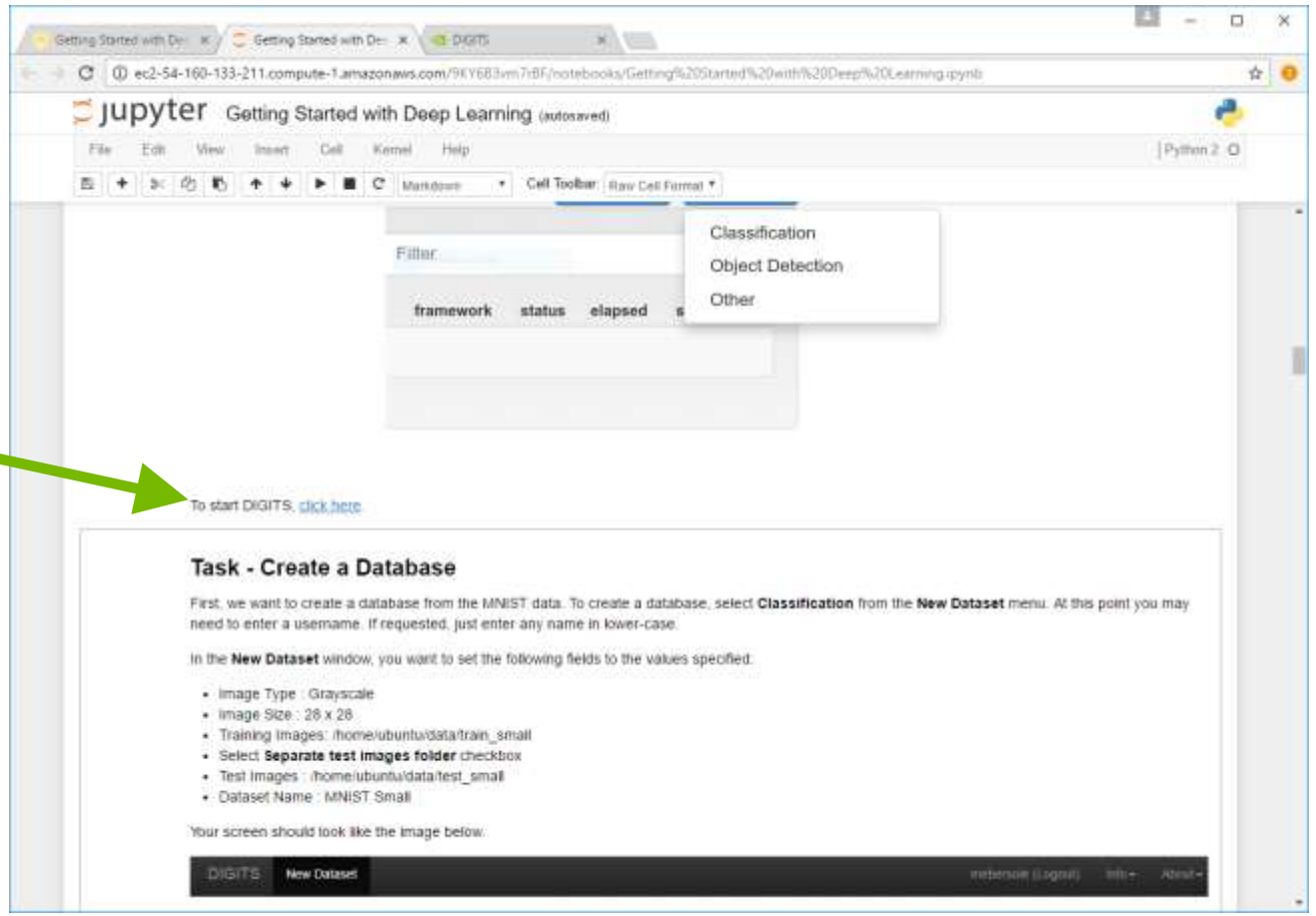


2. Click the "run cell" button

2. Confirm you receive the same result

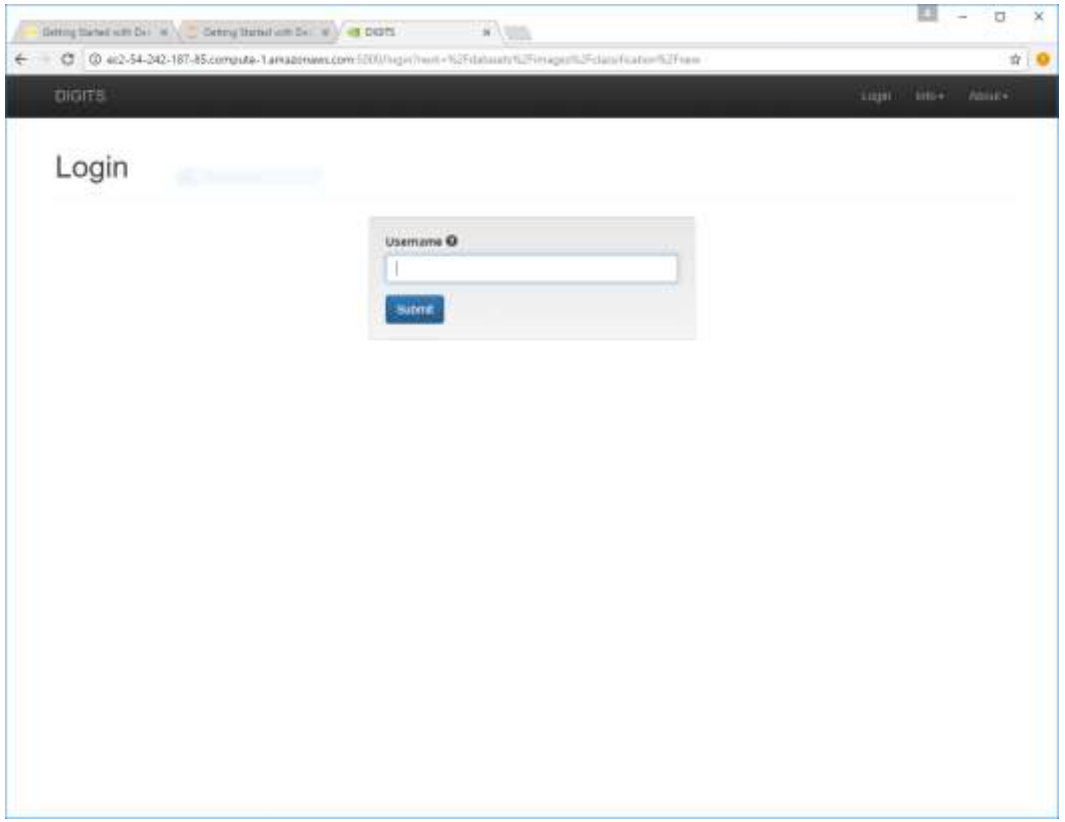
STARTING DIGITS

Instruction in Jupyter notebook will link you to DIGITS



ACCESSING DIGITS

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



LAB DISCUSSION / OVERVIEW

CREATE DATASET IN DIGITS

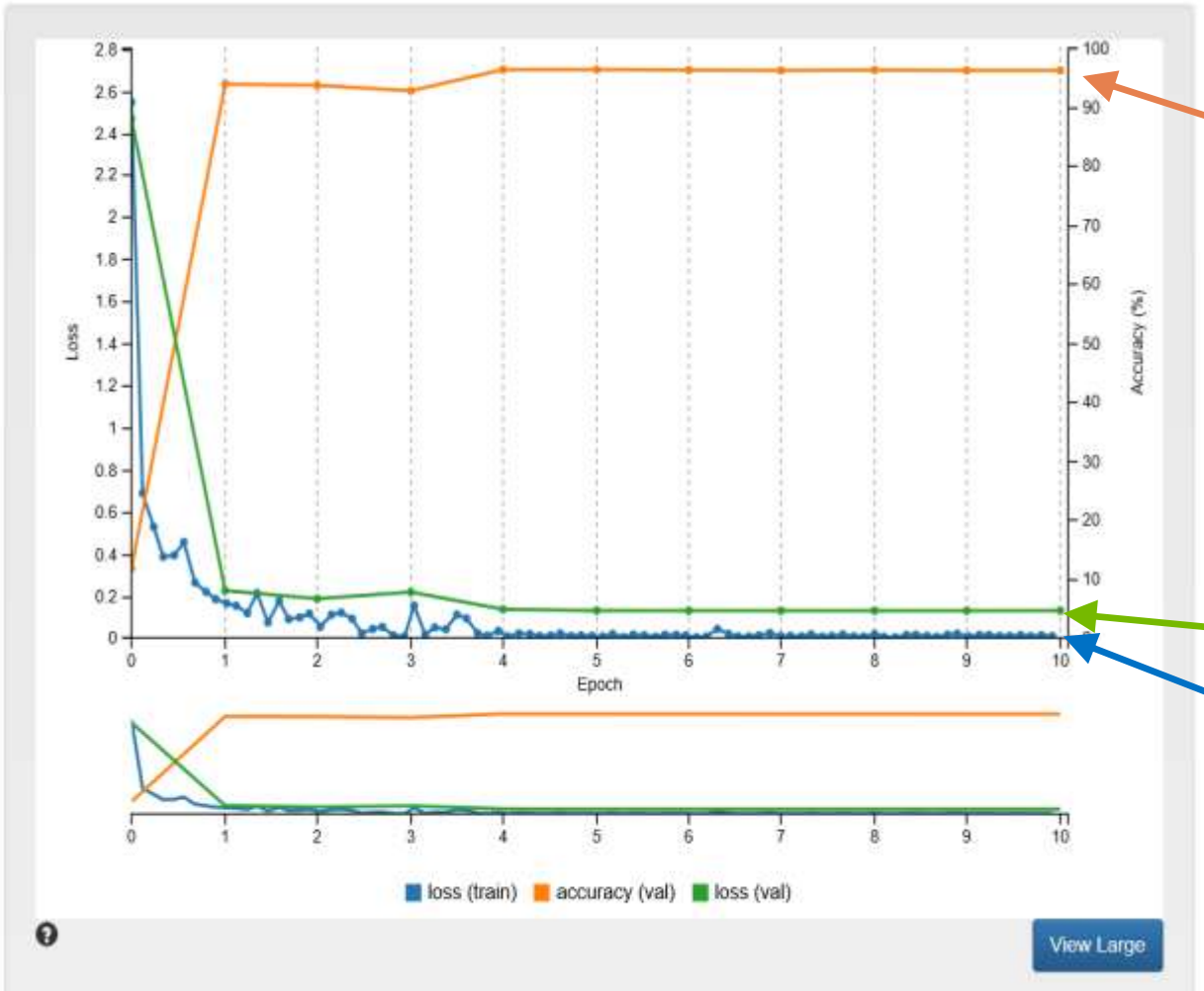
- Dataset settings
  - Image Type: Grayscale
  - Image Size: 28 x 28
  - Training Images: **/home/ubuntu/data/train_small**
  - Select “**Separate test images folder**” checkbox
  - Test Images: **/home/ubuntu/data/test_small**
  - Dataset Name: MNIST Small

CREATE MODEL

- Select the “**MNIST small**” dataset
- Set the number of “**Training Epochs**” to 10
- Set the framework to “**Caffe**”
- Set the model to “**LeNet**”
- Set the name of the model to “**MNIST small**”
- When training done, Classify One :

    /home/ubuntu/data/test_small/2/img_4415.png

EVALUATE THE MODEL



Accuracy obtained from validation dataset

Loss function (Validation)

Loss function (Training)

ADDITIONAL TECHNIQUES TO IMPROVE MODEL

- More training data
- Data augmentation
- Modify the network

LAB REVIEW

FIRST RESULTS

Small dataset (10 epochs)

- 96% of accuracy achieved
- Training is done within one minute

	SMALL DATASET
	1 : 99.90 %
	2 : 69.03 %
	8 : 71.37 %
	8 : 85.07 %
	0 : 99.00 %
	8 : 99.69 %
	8 : 54.75 %

FULL DATASET

6x larger dataset

- Dataset
  - Training Images: /home/ubuntu/data/train_full
  - Test Image: /home/ubuntu/data/test_full
  - Dataset Name: MNIST full
- Model
  - Clone “MNIST small”.
  - Give a new name “MNIST full” to push the create button

SECOND RESULTS

Full dataset (10 epochs)

- 99% of accuracy achieved
- No improvements in recognizing real-world images

	SMALL DATASET	FULL DATASET
	1 : 99.90 %	0 : 93.11 %
	2 : 69.03 %	2 : 87.23 %
	8 : 71.37 %	8 : 71.60 %
	8 : 85.07 %	8 : 79.72 %
	0 : 99.00 %	0 : 95.82 %
	8 : 99.69 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %

DATA AUGMENTATION

Adding Inverted Images

- Pixel(Inverted) = 255 – Pixel(original)
- White letter with black background
  - Black letter with white background
- Training Images: /home/ubuntu/data/train_invert
- Test Image: /home/ubuntu/data/test_invert
- Dataset Name: MNIST invert

DATA AUGMENTATION

Adding inverted images (10 epochs)

	SMALL DATASET	FULL DATASET	+INVERTED
	1 : 99.90 %	0 : 93.11 %	1 : 90.84 %
	2 : 69.03 %	2 : 87.23 %	2 : 89.44 %
	8 : 71.37 %	8 : 71.60 %	3 : 100.0 %
	8 : 85.07 %	8 : 79.72 %	4 : 100.0 %
	0 : 99.00 %	0 : 95.82 %	7 : 82.84 %
	8 : 99.69 %	8 : 100.0 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %	2 : 96.27 %

MODIFY THE NETWORK

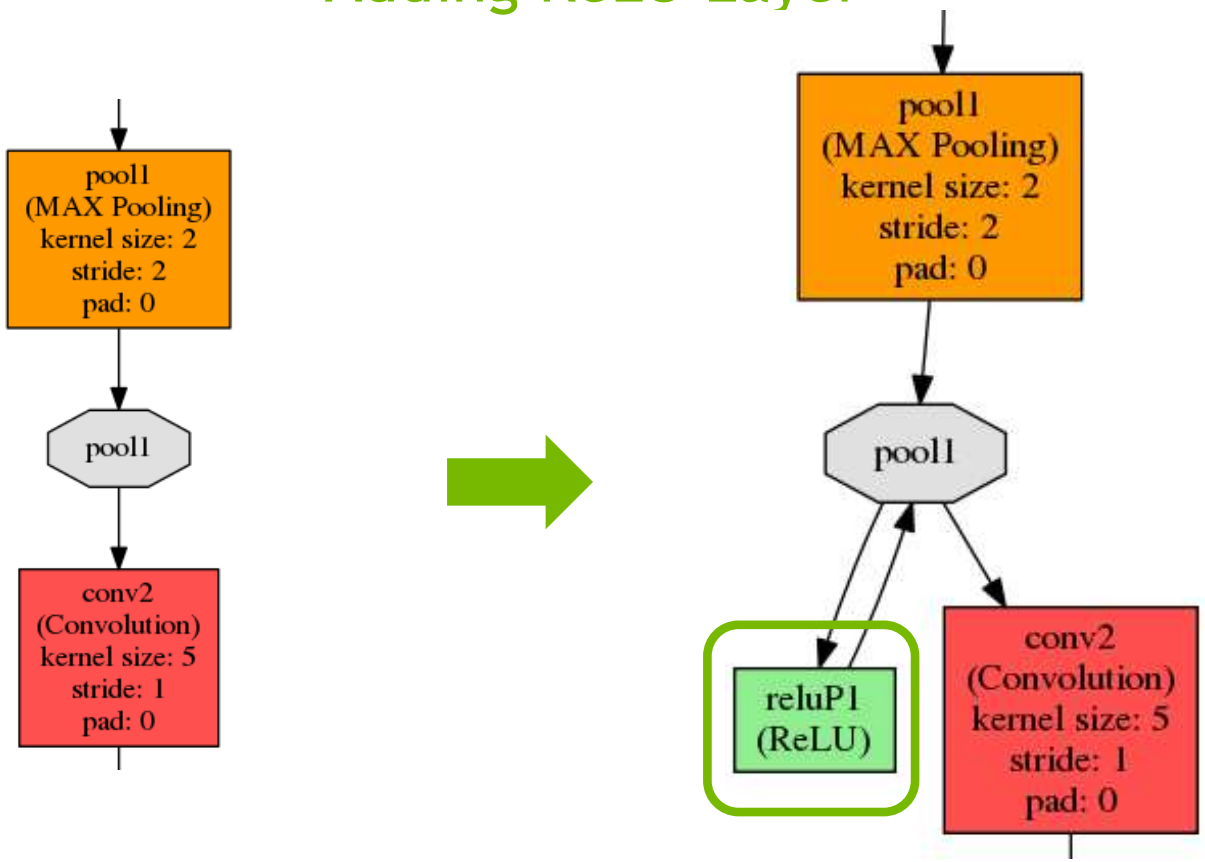
Adding filters and ReLU layer

```
layer {
        name: "pool1“
        type: "Pooling“
        …
}

layer {
        name: "reluP1"
        type: "ReLU"
        bottom: "pool1"
        top: "pool1"
}

layer {
        name: "reluP1“
```

```
layer {
  name: "conv1"
  type: "Convolution"
        ...
        convolution_param {
        num_output: 75
        ...
layer {
        name: "conv2"
        type: "Convolution"
        ...
        convolution_param {
        num_output: 100
        ...
```

MODIFY THE NETWORK

Adding ReLU Layer



MODIFIED NETWORK

Adding filters and ReLU layer (10 epochs)

	SMALL DATASET	FULL DATASET	+INVERTED	ADDING LAYER
	1 : 99.90 %	0 : 93.11 %	1 : 90.84 %	1 : 59.18 %
	2 : 69.03 %	2 : 87.23 %	2 : 89.44 %	2 : 93.39 %
	8 : 71.37 %	8 : 71.60 %	3 : 100.0 %	3 : 100.0 %
	8 : 85.07 %	8 : 79.72 %	4 : 100.0 %	4 : 100.0 %
	0 : 99.00 %	0 : 95.82 %	7 : 82.84 %	2 : 62.52 %
	8 : 99.69 %	8 : 100.0 %	8 : 100.0 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %	2 : 96.27 %	8 : 70.83 %

WHAT'S NEXT

- Use / practice what you learned
- Discuss with peers practical applications of DNN
- Reach out to NVIDIA and the Deep Learning Institute

www.nvidia.com/dli