



DEEP  
LEARNING  
INSTITUTE

# Image Classification with DIGITS

---

Hyungon Ryu

Certified Instructor, NVIDIA Deep Learning Institute  
NVIDIA Corporation



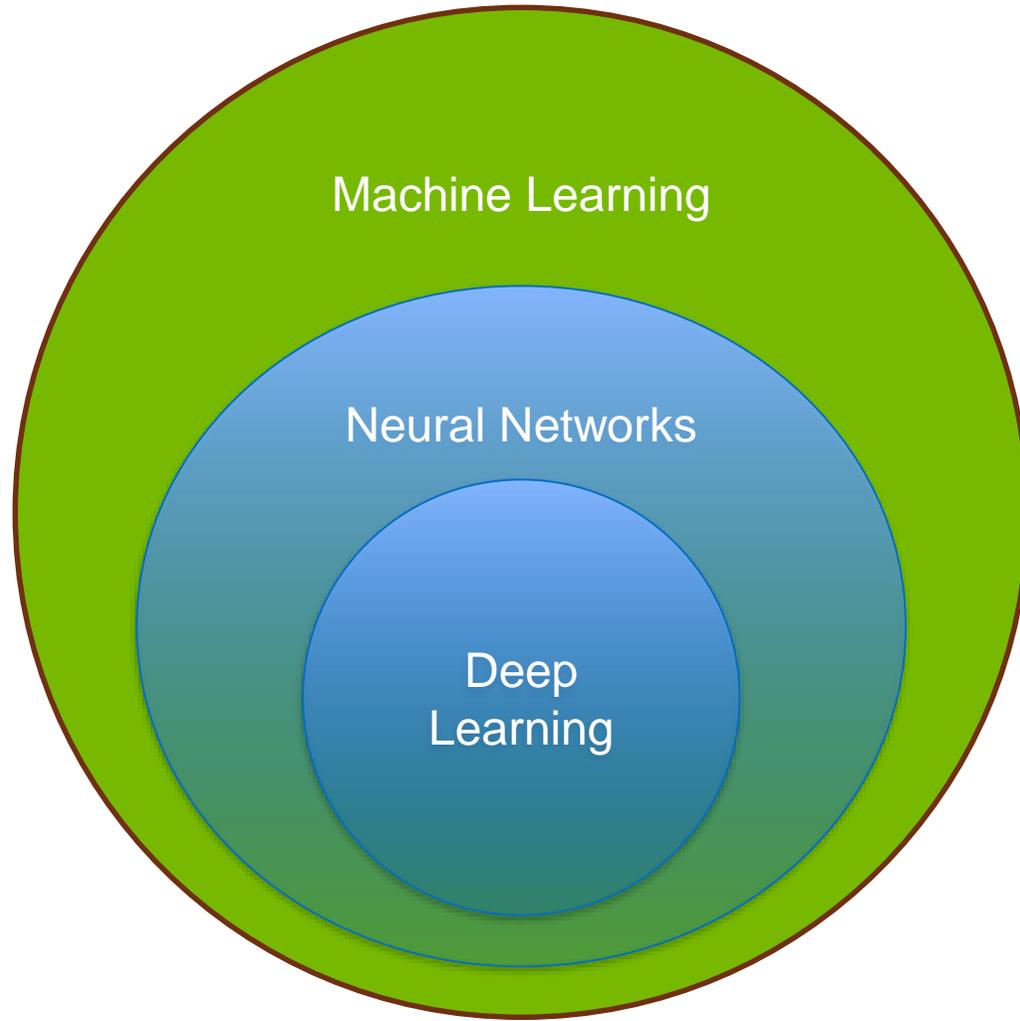
# DEEP LEARNING INSTITUTE

## DLI Mission

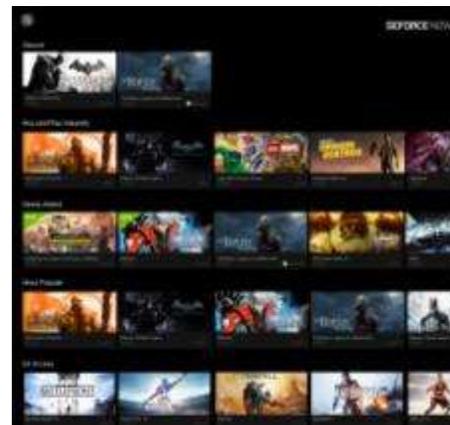
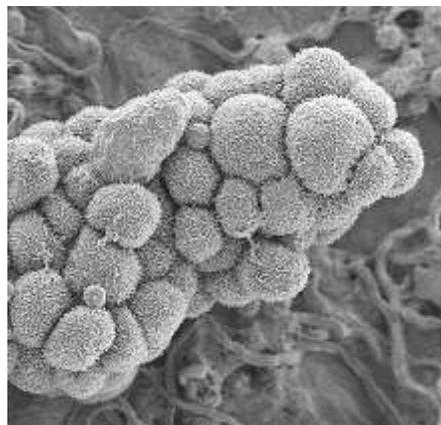
Helping people solve challenging problems using AI and deep learning.

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

**WHAT IS DEEP LEARNING?**



# 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

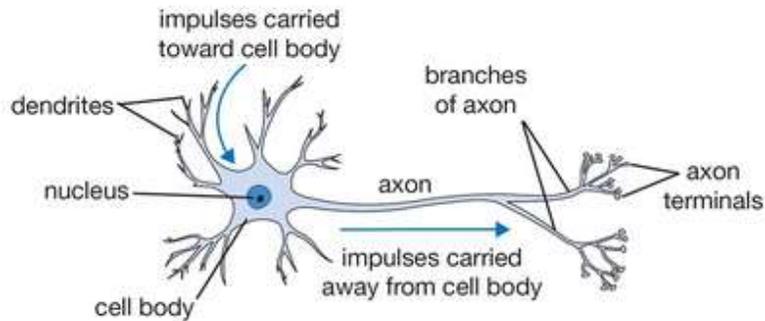
Face Detection  
Video Surveillance  
Satellite Imagery

## AUTONOMOUS MACHINES

Pedestrian Detection  
Lane Tracking  
Recognize Traffic Sign

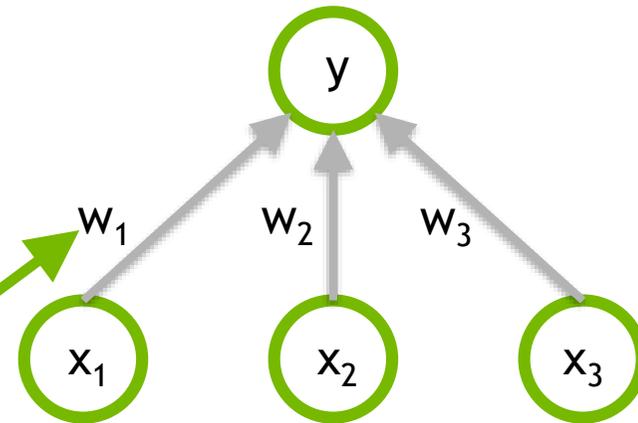
# ARTIFICIAL NEURONS

Biological neuron



From Stanford cs231n lecture notes

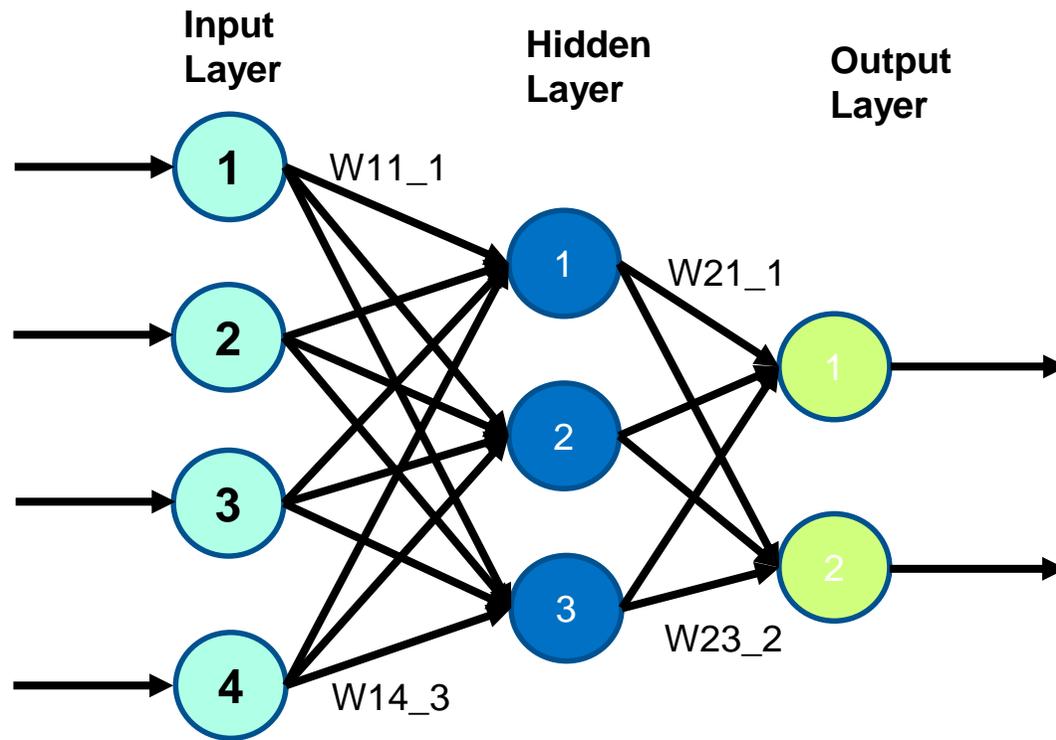
Artificial neuron



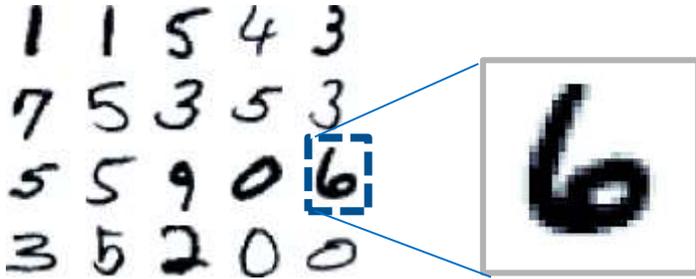
Weights ( $w_n$ )  
= parameters

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

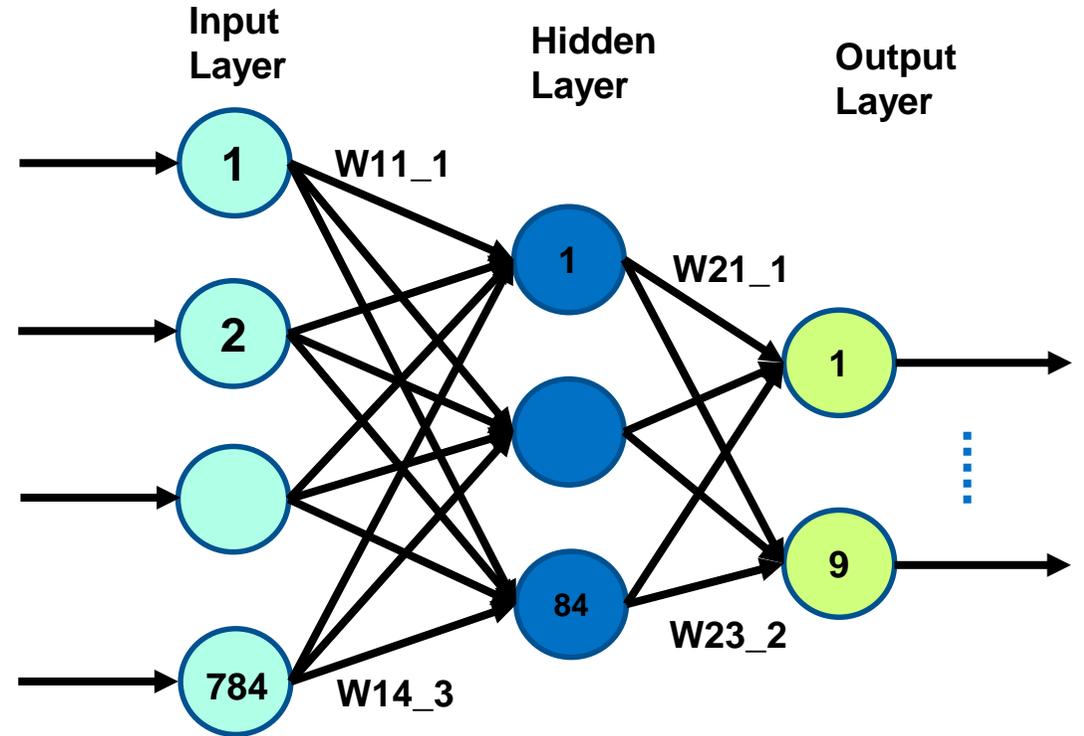
# MLP



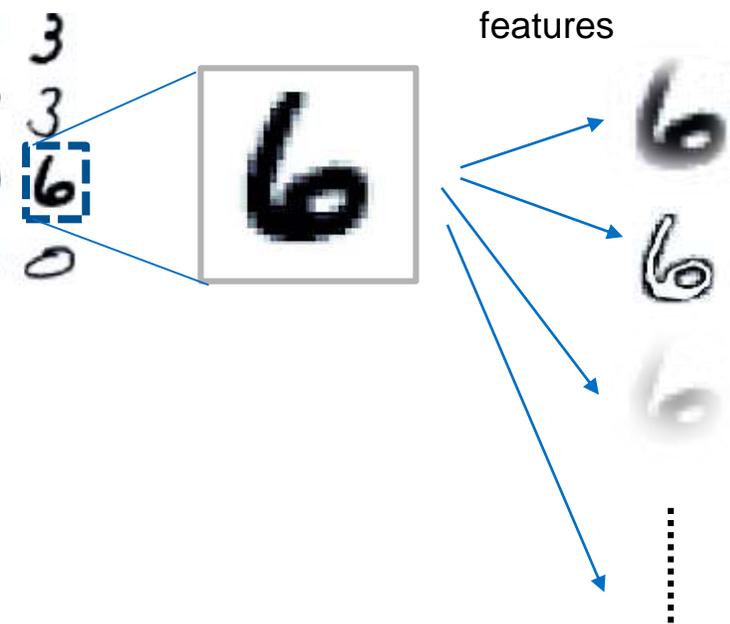
# ANN for MNIST



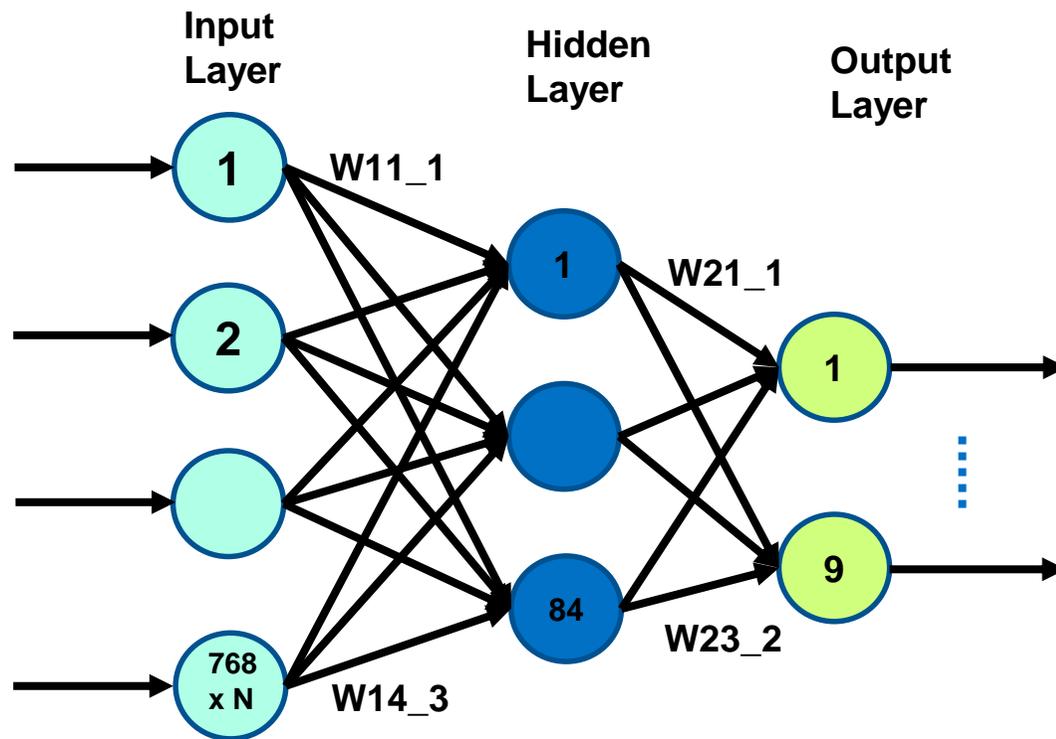
$28 \times 28 = 784$



# Pre-processing + ANN for MNIST



768 x N



# Feature with Convolution Filter



edge

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$


blur

$$\begin{bmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{bmatrix}$$


Motion blur

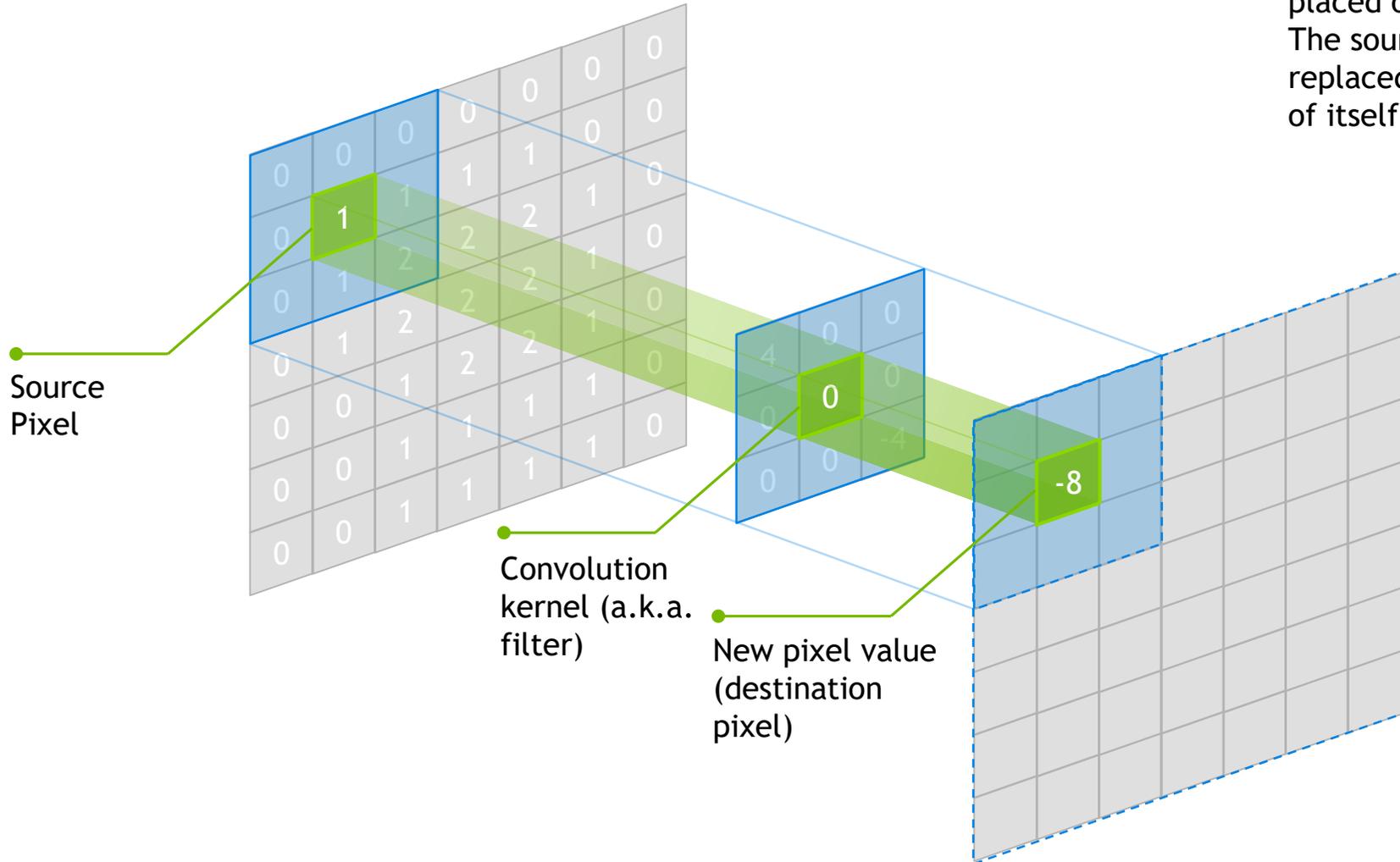
$$\begin{bmatrix} 0 & 0 & 0.3 \\ 0 & 0.3 & 0 \\ 0.3 & 0 & 0 \end{bmatrix}$$

sharpen

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

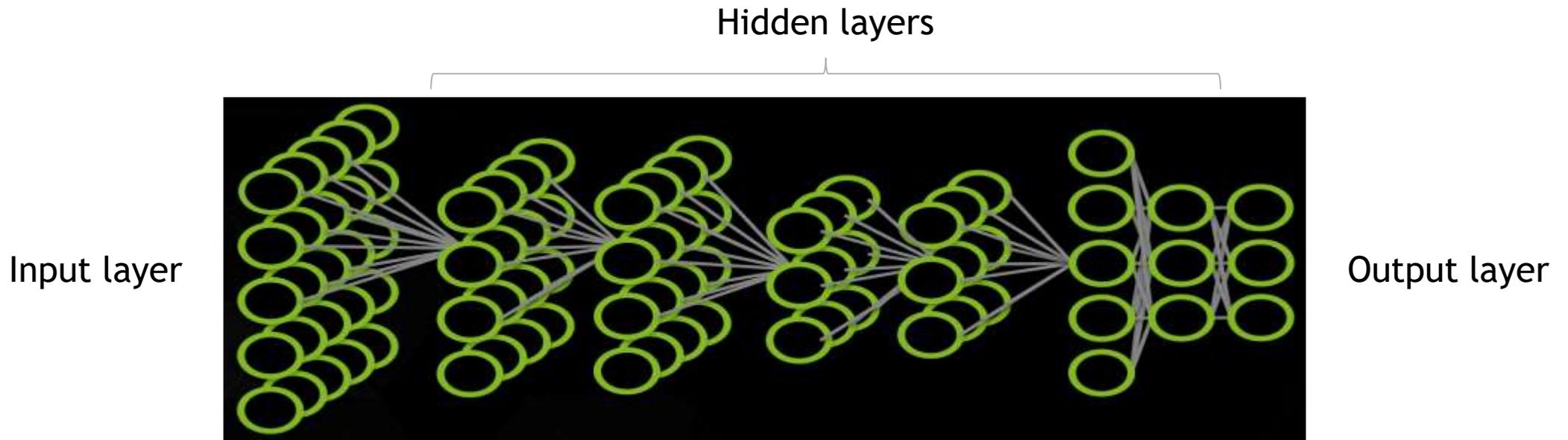
# CONVOLUTION

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.



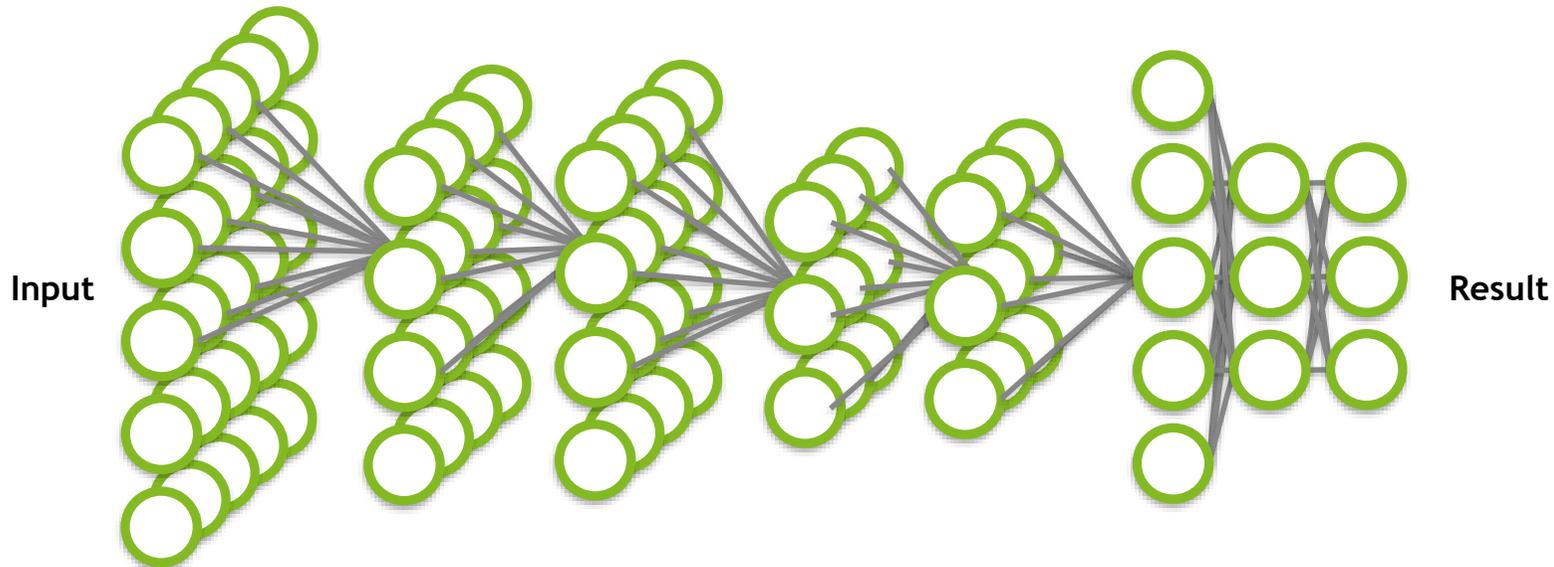
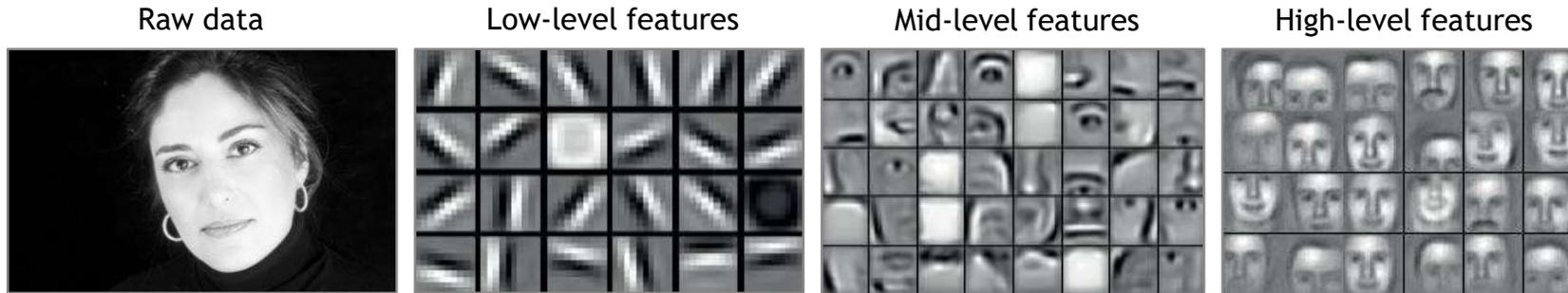
# ARTIFICIAL NEURAL NETWORK

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



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

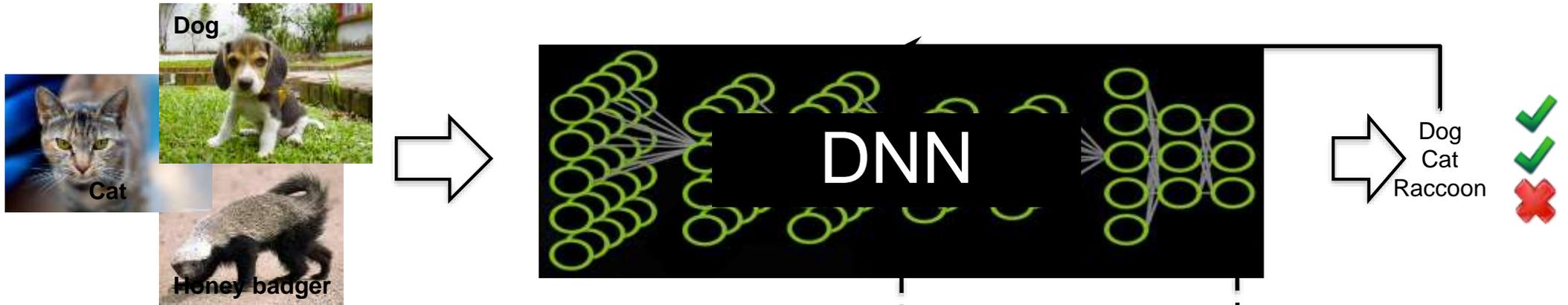
**Training data**  
10-100M images

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

**Learning algorithm**  
~30 Exaflops  
1-30 GPU days

# DEEP LEARNING APPROACH

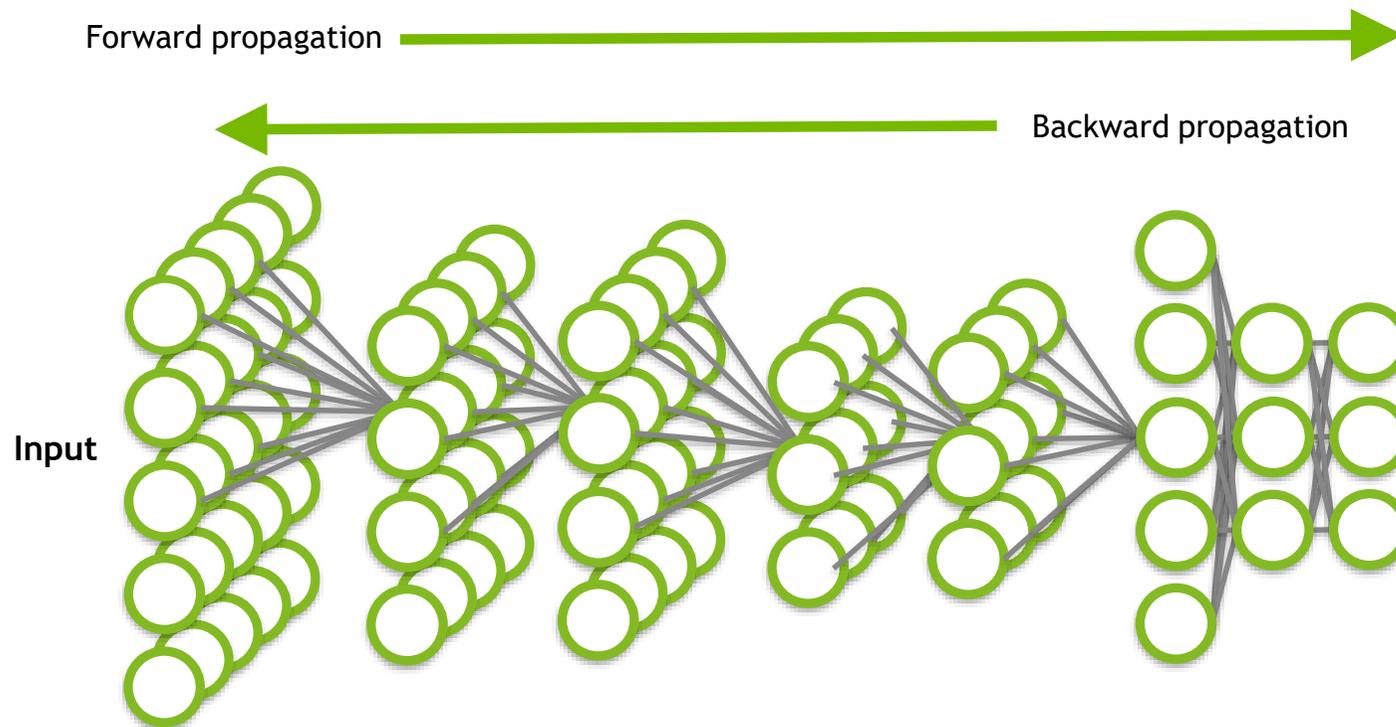
Train:



Deploy:



# 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

OBJECT  
DETECTION



IMAGE  
CLASSIFICATION

DL4J  
Deeplearning4j



## SPEECH & AUDIO

VOICE  
RECOGNITION



MINERVA

LANGUAGE  
TRANSLATION



## NATURAL LANGUAGE PROCESSING

RECOMMENDATION  
ENGINES



Pylearn2

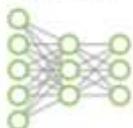
SENTIMENT  
ANALYSIS



theano

## NVIDIA DEEP LEARNING SDK

cuDNN



TensorRT



DeepStream SDK



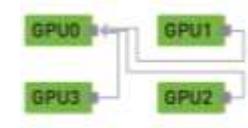
cuBLAS



cuSPARSE



NCCL



# WHAT IS CAFFE?

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



- 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.berkeleyvision.org](http://caffe.berkeleyvision.org)  
<http://github.com/BVLC/caffe>

# 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

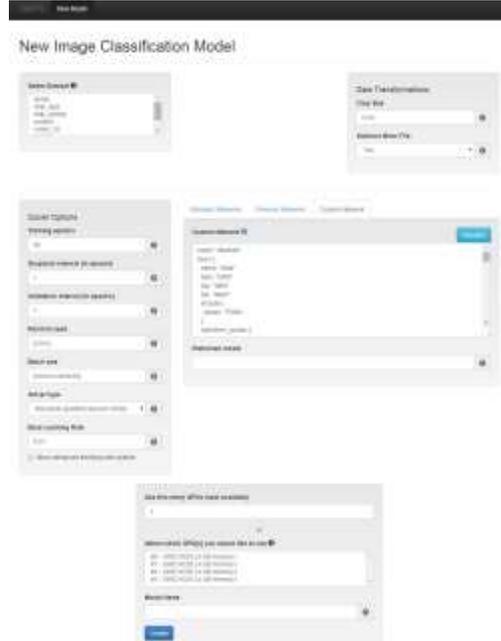
# NVIDIA DIGITS

## Interactive Deep Learning GPU Training System

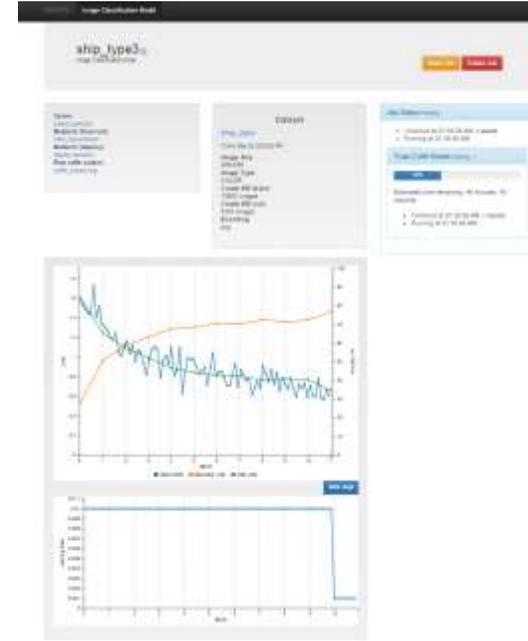
### Process Data



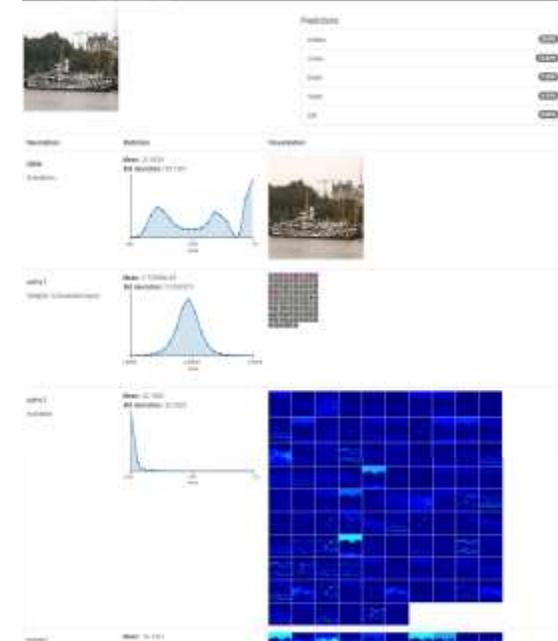
### Configure DNN



### 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

The screenshot shows the DIGITS Home interface. At the top, a dark navigation bar contains the text 'DIGITS' on the left, and 'ckillam (Logout)', 'Info', and 'About' on the right. Below this, the word 'Home' is displayed in a large font. To the right of 'Home', the text '1/1 GPU available' is shown. The main content area features a status indicator 'No Jobs Running' and a row of buttons: 'Datasets (0)', 'Models (0)', 'Pretrained Models (0)', and a 'Rectangular Snip' button. Below these buttons, there is a section for 'Group Jobs' with a checked checkbox. This section includes 'Delete' and 'Group' buttons, a search bar labeled 'Filter', and a table with columns for 'framework', 'status', 'elapsed', and 'submitted'. The table currently shows 'No Models'. Three green circles highlight specific elements: one around 'DIGITS', one around the 'Datasets (0)' and 'Models (0)' buttons, and one around the 'New Model' button which has a dropdown menu showing 'Images'.

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

## New Object Detection Dataset

**Object Detection Dataset Options**

Images can be stored in any of the supported file formats (.png, .jpg, .jpeg, .bmp, .ppm).

**Training image folder**

Label files are expected to have the .txt extension. For example if an image file is named foo.png the corresponding label file should be foo.txt.

**Training label folder**

**Validation image folder**

**Validation label folder**

**Pad image (Width x Height)**

 x 

**Resize image (Width x Height)**

 x 

**Channel conversion**

**Minimum box size (in pixels) for validation set**

**Custom classes**

## New Image Classification Dataset

**Image Type**

**Image size (Width x Height)**

 x 

**Resize Transformation**

[See example](#)

Use Image Folder  Use Text Files

**Training images**

**Minimum samples per class**

**Maximum samples per class**

**% for validation**

**% for testing**

Separate validation images folder  
 Separate test images folder

**DB backend**

**Image Encoding**

**Group Name**

**Dataset Name**

[Create](#)

Different options will be presented based upon the task

# DIGITS - MODEL

## New Object Detection Model

Select Dataset

Python Layers

Server-side file

Use client-side file

Solver Options

Training epochs

30

Snapshot interval (in epochs)

1

Validation interval (in epochs)

1

Random seed

[none]

Batch size

[network defaults]

Batch Accumulation

Solver type

Stochastic gradient descent (SGD)

Base Learning Rate

0.01

Show advanced learning rate options

Data Transformations

Subtract Mean

image

Crop Size

none

## New Image Classification Model

Select Dataset

Python Layers

Server-side file

Use client-side file

Solver Options

Training epochs

30

Snapshot interval (in epochs)

1

Validation interval (in epochs)

1

Random seed

[none]

Batch size

[network defaults]

Batch Accumulation

Solver type

Stochastic gradient descent (SGD)

Base Learning Rate

0.01

Show advanced learning rate options

Data Transformations

Subtract Mean

image

Crop Size

none

Define custom layers with Python

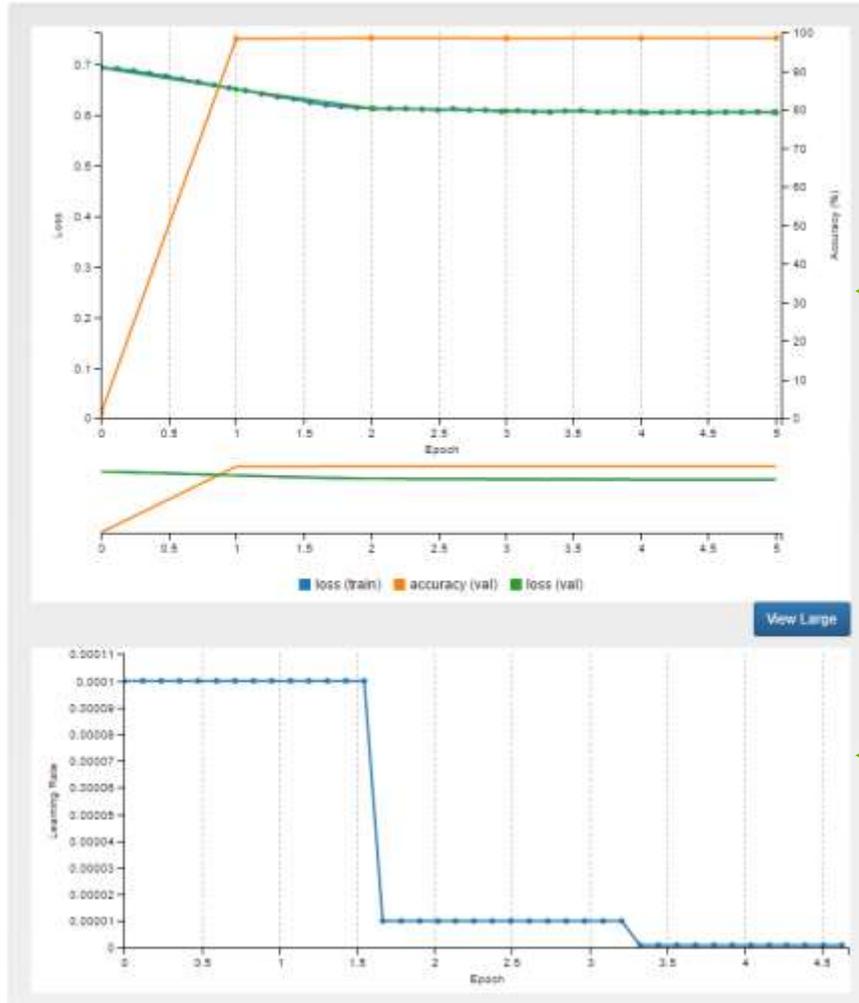
Can anneal the learning rate

Network	Details	Intended image size
LeNet	Original paper (1998)	28x28 (gray)

Network	Details	Intended image size
LeNet	Original paper (1998)	28x28 (gray)

Differences may exist between model tasks

# DIGITS - TRAINING

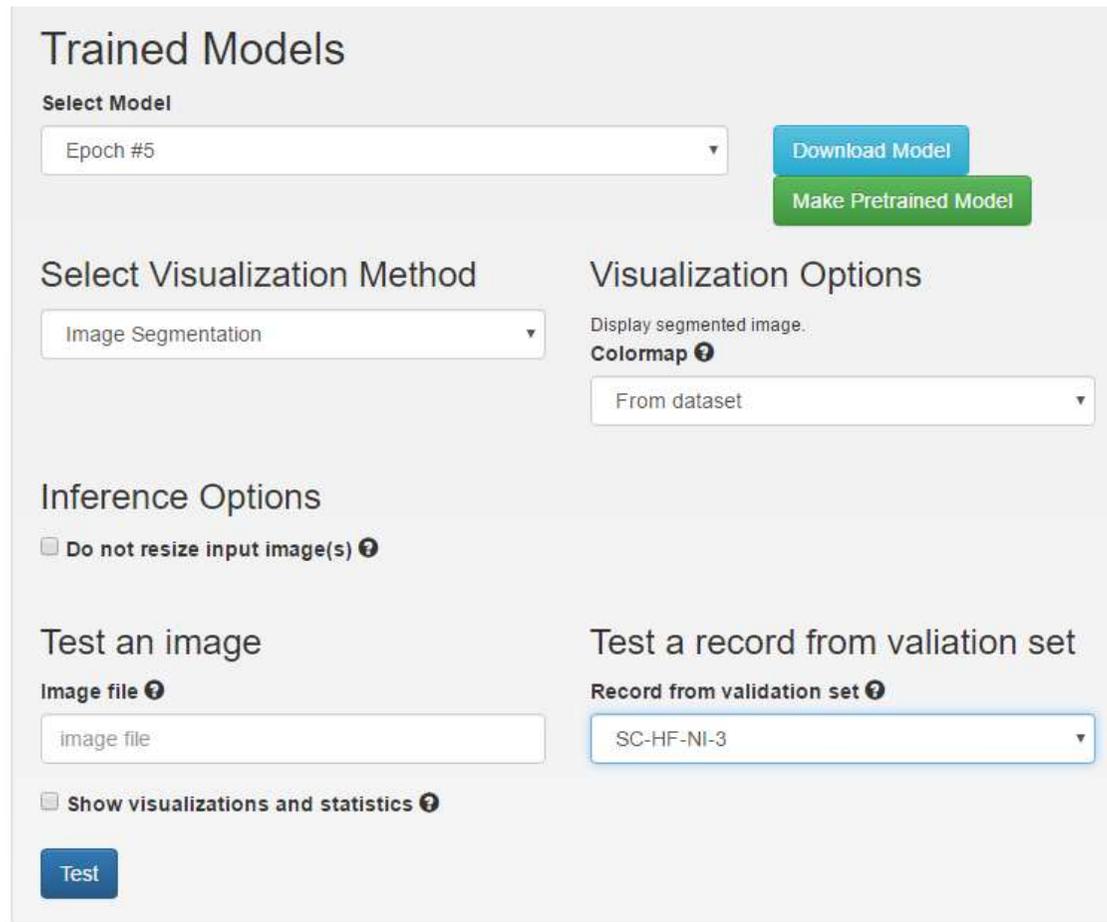


Loss function and accuracy during training

Annealed learning rate

# DIGITS - VISUALIZATION

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

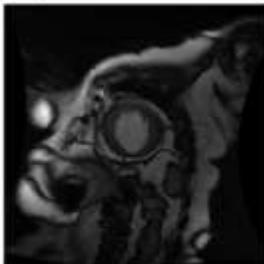


The screenshot shows the 'Trained Models' section of the DIGITS interface. It includes a 'Select Model' dropdown menu with 'Epoch #5' selected, and two buttons: 'Download Model' (blue) and 'Make Pretrained Model' (green). Below this is the 'Select Visualization Method' dropdown menu with 'Image Segmentation' selected. To the right, under 'Visualization Options', there is a checkbox for 'Display segmented image.' and a 'Colormap' dropdown menu with 'From dataset' selected. In the 'Inference Options' section, there is a checkbox for 'Do not resize input image(s)'. The 'Test an image' section has an 'Image file' input field with 'image file' entered. The 'Test a record from validation set' section has a 'Record from validation set' dropdown menu with 'SC-HF-NI-3' selected. At the bottom, there is a checkbox for 'Show visualizations and statistics' and a 'Test' button.

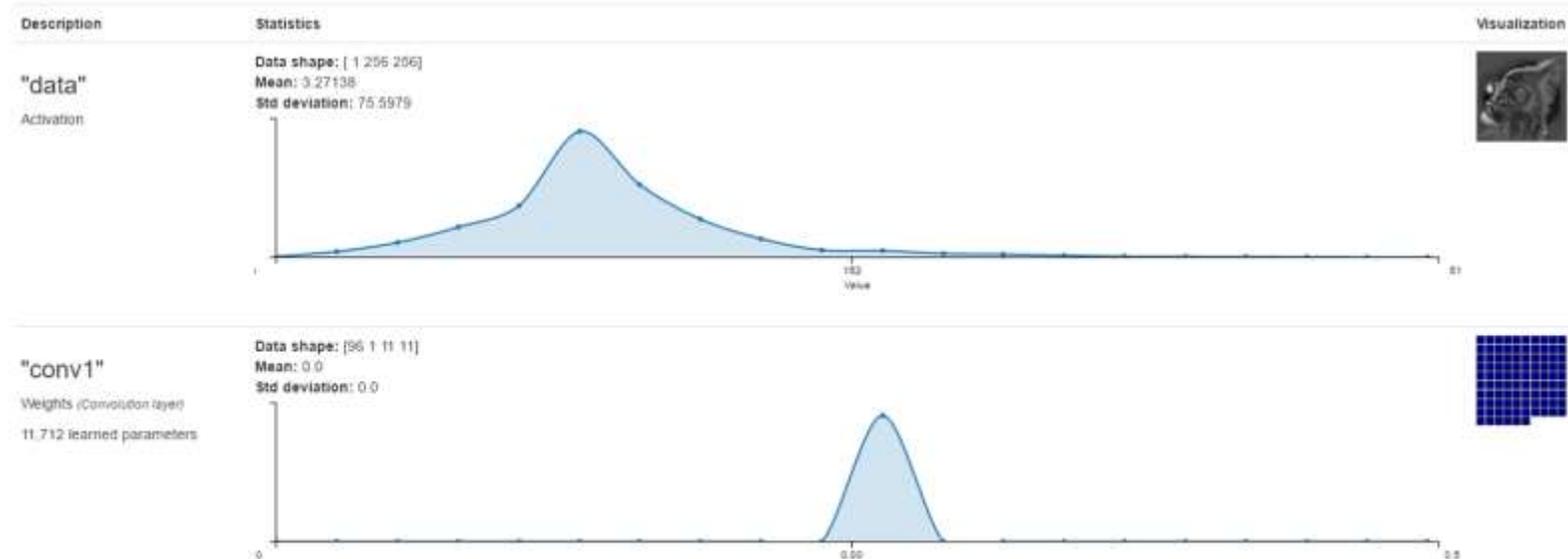
# DIGITS - VISUALIZATION RESULTS

## Summary

### Output visualizations



### Layer visualizations

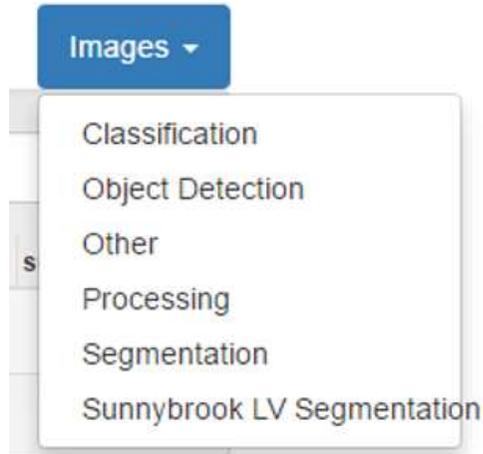


# DIGITS PLUGINS

## DIGITS Plugins

Image : Sunnybrook LV Segmentation

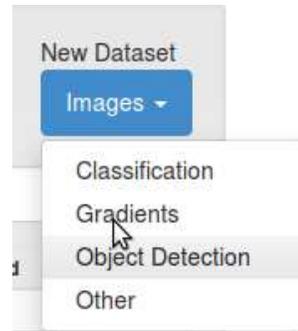
plugins/data/sunnybrook



## DIGITS Plugins

Image : Regression

plugins/data/imageGradients



## DIGITS Plugins

Text

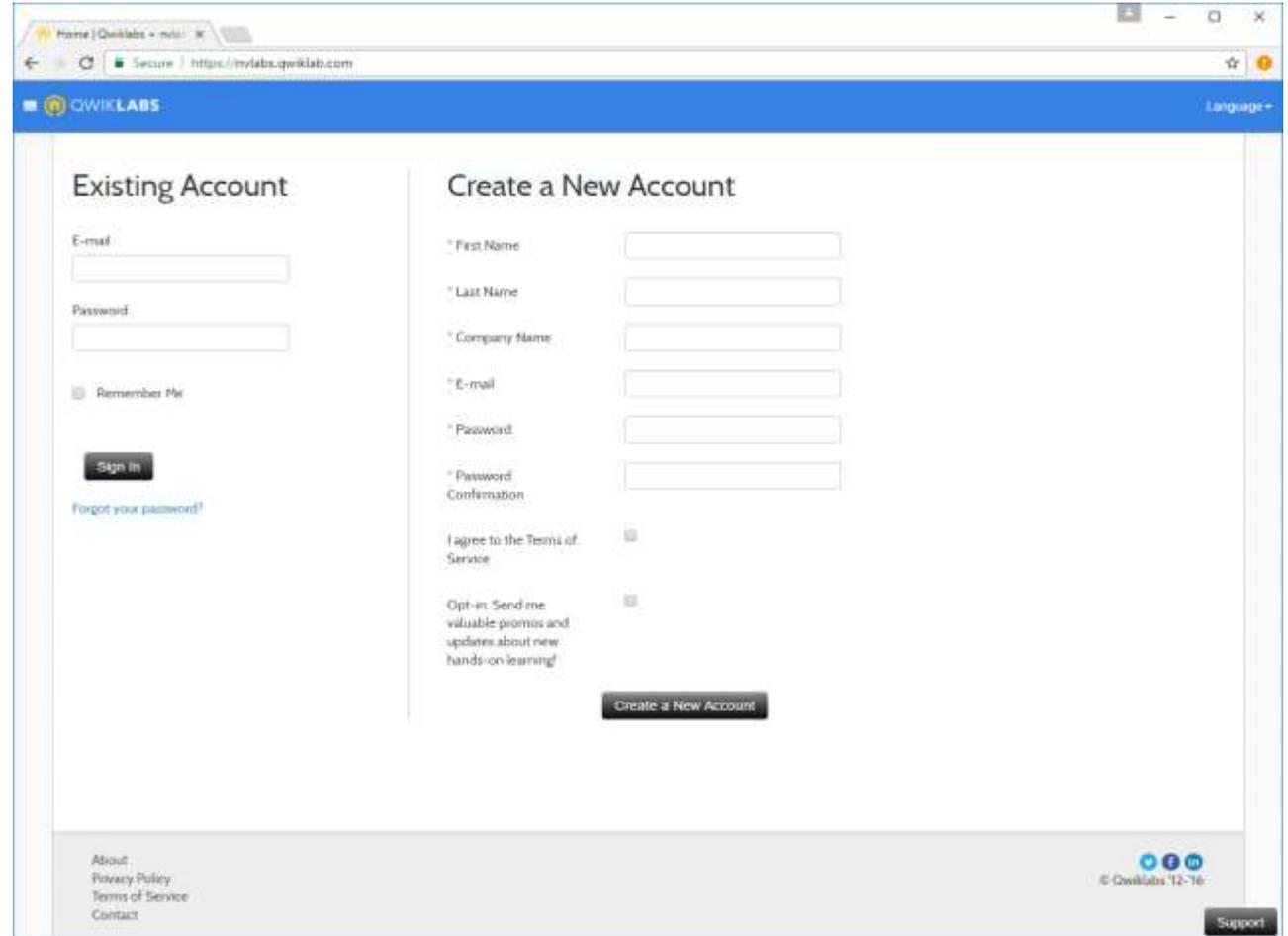
plugins/data/textClassification



# LAUNCHING THE LAB ENVIRONMENT

# NAVIGATING TO QWIKLABS

1. Navigate to:  
<https://nvlabs.qwiklab.com>
1. Login or create a new account



The screenshot shows the Qwiklabs website interface. The browser address bar displays "https://nvlabs.qwiklab.com". The page features a blue header with the Qwiklabs logo and a "Language" dropdown. The main content area is divided into two columns: "Existing Account" and "Create a New Account".

**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!
- 

**Footer:**

- Links: About, Privacy Policy, Terms of Service, Contact
- © Qwiklabs 12-16
- Support:

# 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

In-Session Class: GTC2017

125.3 Total Hours

68 Completed Labs

8 Classes Taken

### Class Details

- Deep Learning for Image Segmentation
- Neural Network Deployment with DIGITS and TensorRT
- Image Classification with DIGITS**
- Medical Image Segmentation Using DIGITS
- Object Detection with DIGITS
- Photo Editing with Generative Adversarial Networks in Tensorflow and DIGITS
- Accelerating Applications with CUDA C/C++

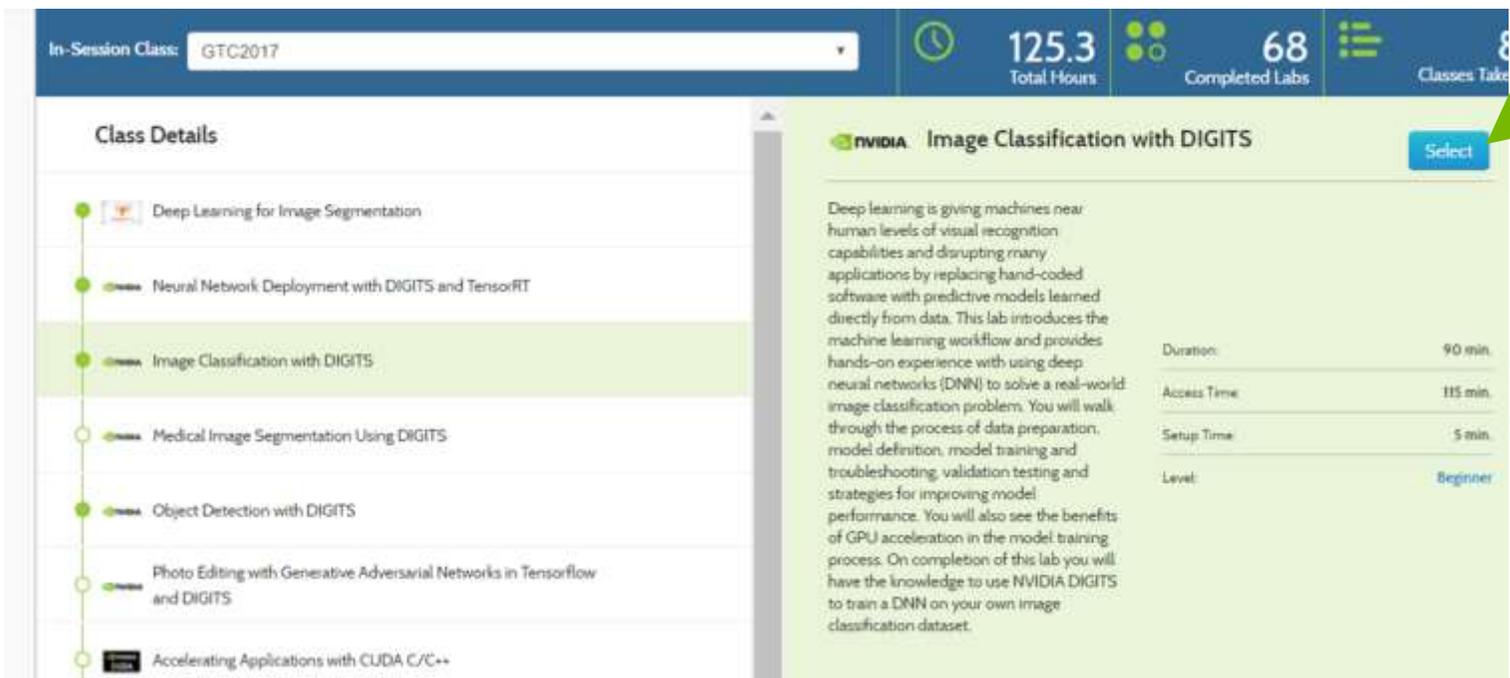
### NVIDIA Image Classification with DIGITS

Deep learning is giving machines near human levels of visual recognition capabilities and disrupting many applications by replacing hand-coded software with predictive models learned directly from data. This lab introduces the machine learning workflow and provides hands-on experience with using deep neural networks (DNN) to solve a real-world image classification problem. You will walk through the process of data preparation, model definition, model training and troubleshooting, validation testing and strategies for improving model performance. You will also see the benefits of GPU acceleration in the model training process. On completion of this lab you will have the knowledge to use NVIDIA DIGITS to train a DNN on your own image classification dataset.

Duration:	90 min.
Access Time:	115 min.
Setup Time:	5 min.
Level:	Beginner

Select

# LAUNCHING THE LAB ENVIRONMENT



In-Session Class: GTC2017

125.3 Total Hours

68 Completed Labs

8 Classes Taken

### Class Details

- Deep Learning for Image Segmentation
- Neural Network Deployment with DIGITS and TensorRT
- Image Classification with DIGITS**
- Medical Image Segmentation Using DIGITS
- Object Detection with DIGITS
- Photo Editing with Generative Adversarial Networks in Tensorflow and DIGITS
- Accelerating Applications with CUDA C/C++

### nvidia Image Classification with DIGITS

Deep learning is giving machines near human levels of visual recognition capabilities and disrupting many applications by replacing hand-coded software with predictive models learned directly from data. This lab introduces the machine learning workflow and provides hands-on experience with using deep neural networks (DNN) to solve a real-world image classification problem. You will walk through the process of data preparation, model definition, model training and troubleshooting, validation testing and strategies for improving model performance. You will also see the benefits of GPU acceleration in the model training process. On completion of this lab you will have the knowledge to use NVIDIA DIGITS to train a DNN on your own image classification dataset.

Duration: 90 min

Access Time: 115 min

Setup Time: 5 min

Level: Beginner

Select

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

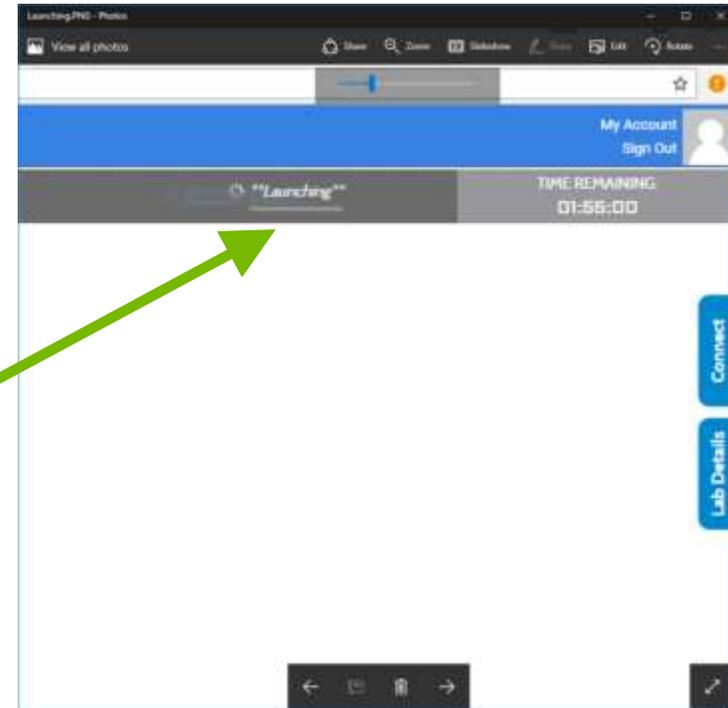
- After a short wait, lab Connection information will be shown
- Please ask Lab Assistants for help!

# 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

**Connect**

Lab Connection  
Please follow the lab instructions to connect to your lab

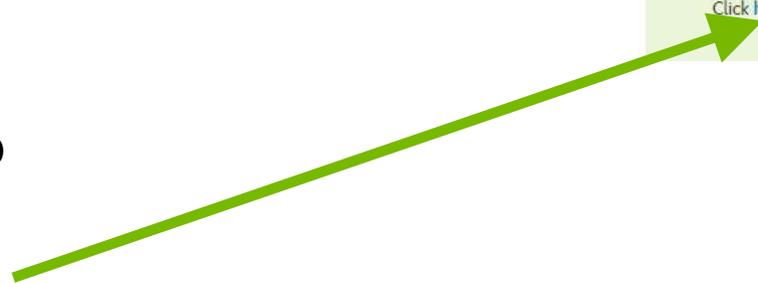
Warning: Do not transmit data into the AWS Console that is not related to Qwiklabs or the lab you are taking.

**Custom Connection Details**

Click [here](#) to launch your lab.

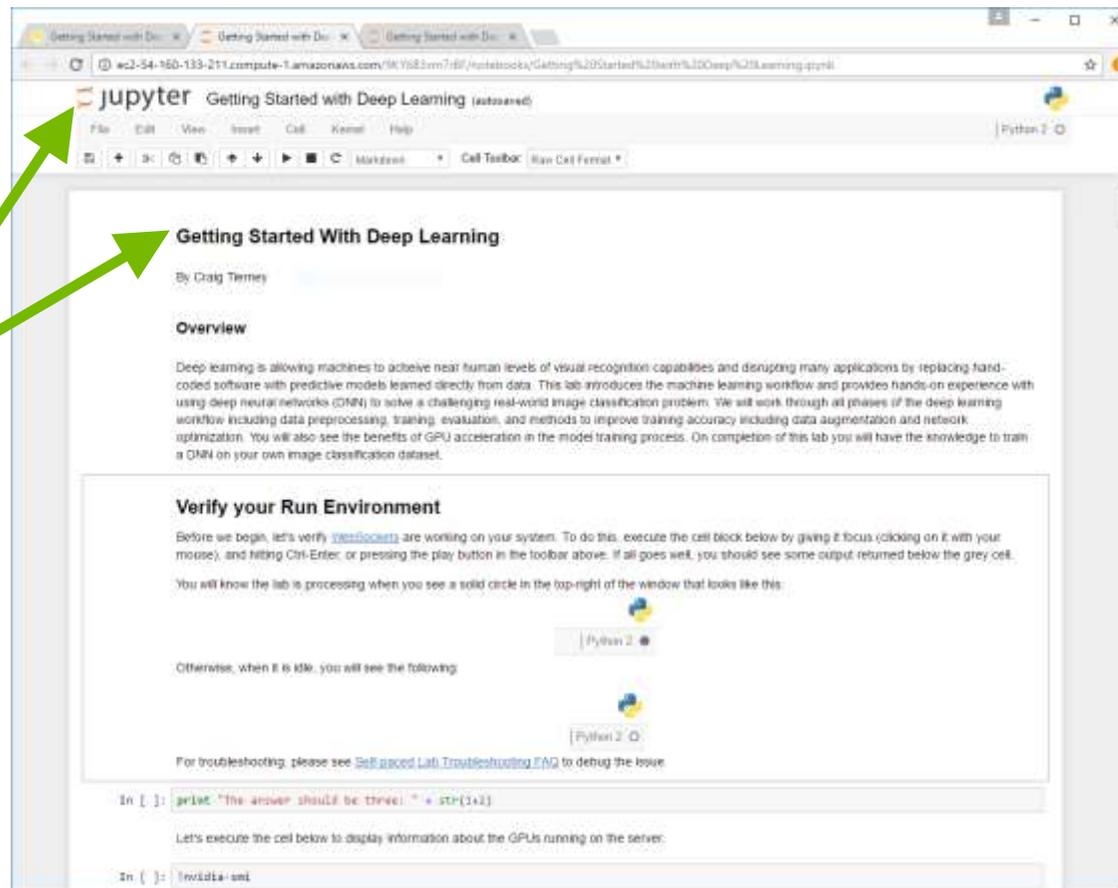
**Lab Details**

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



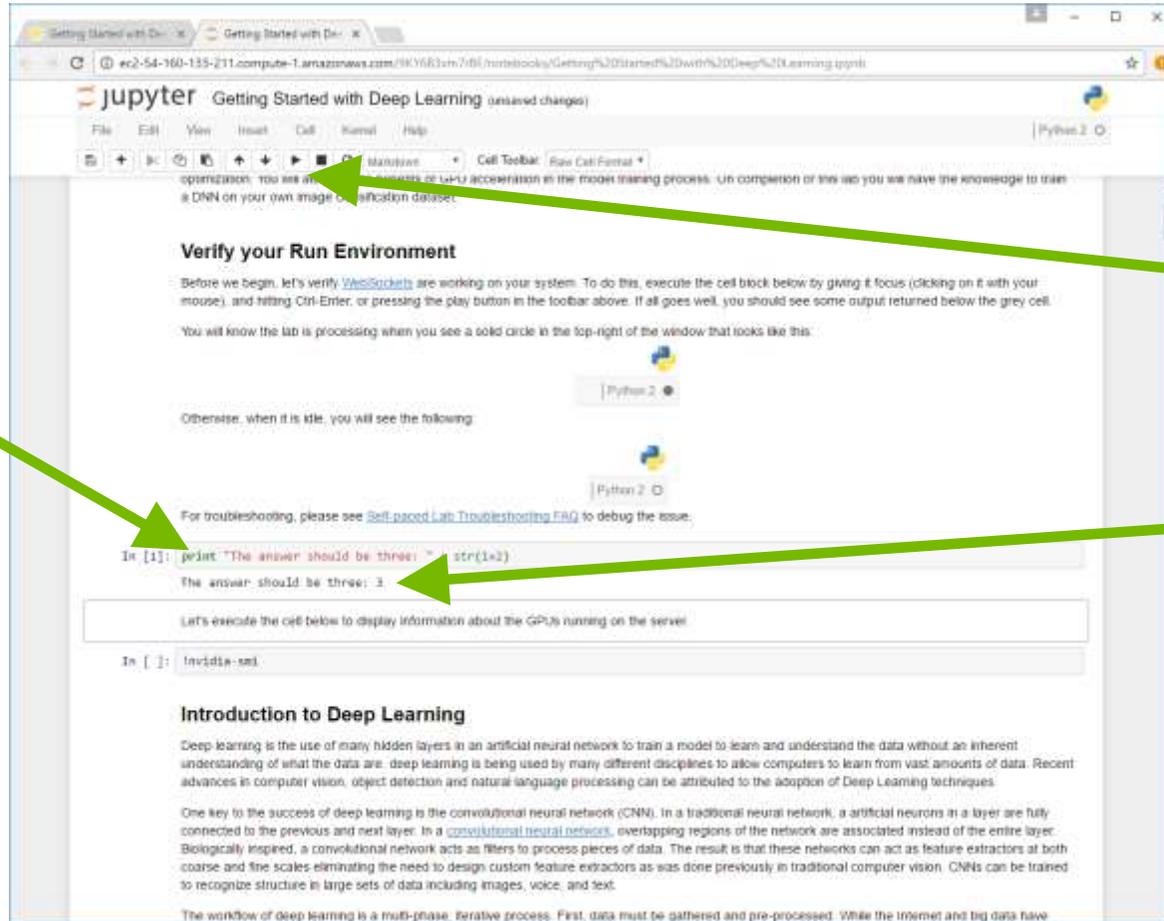
# CONNECTING TO THE LAB ENVIRONMENT

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



# JUPYTER NOTEBOOK

1. Place your cursor in the code

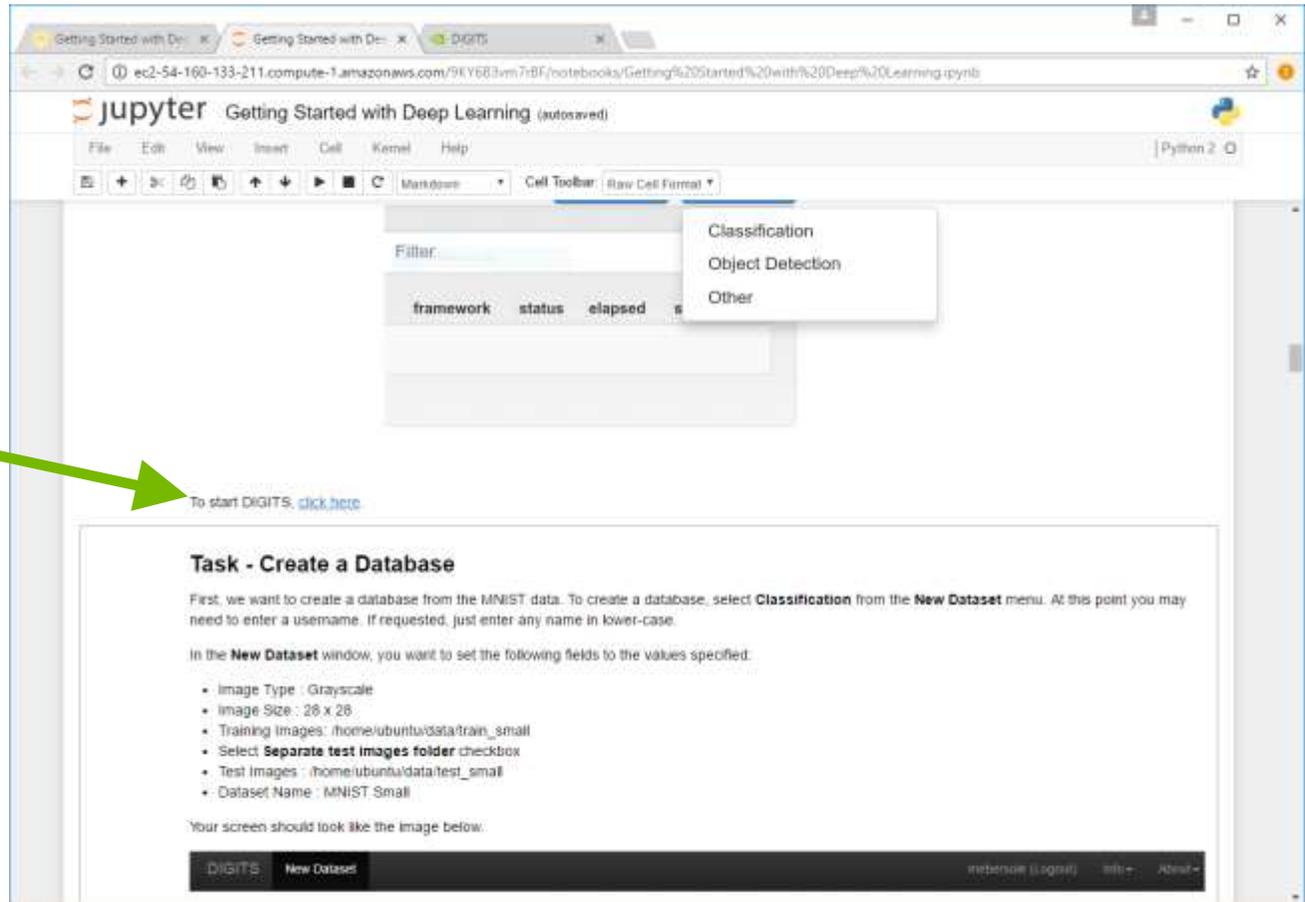


2. Click the "run cell" button

2. Confirm you receive the same result

# STARTING DIGITS

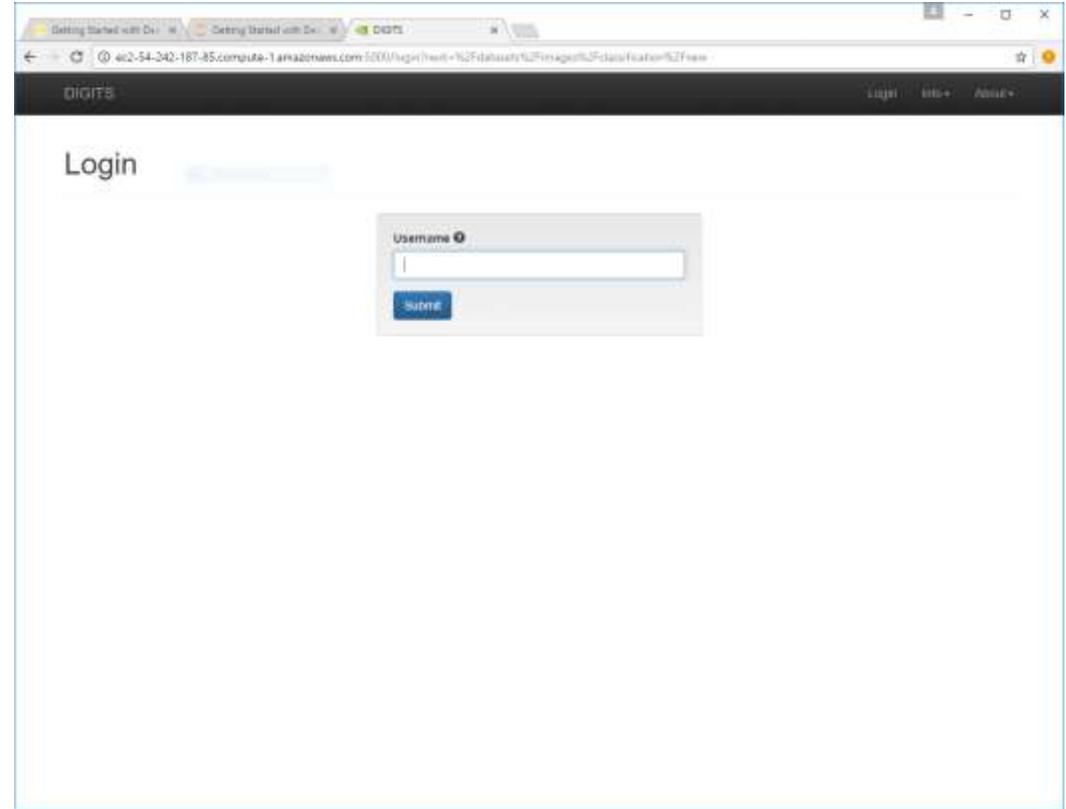
Instruction in Jupyter notebook will link you to DIGITS



The screenshot shows a Jupyter notebook interface with the title "Getting Started with Deep Learning (autosaved)". The "New Dataset" menu is open, showing options: "Classification", "Object Detection", and "Other". The "Classification" option is selected. Below the menu, there is a table with columns "framework", "status", and "elapsed". In the notebook content, there is a link "click here" that is highlighted by a green arrow pointing from the text "Instruction in Jupyter notebook will link you to DIGITS". Below the link, there is a section titled "Task - Create a Database" with instructions on how to create a database from the MNIST data. The instructions include: "First, we want to create a database from the MNIST data. To create a database, select **Classification** from the **New Dataset** menu. At this point you may need to enter a username. If requested, just enter any name in lower-case." and "In the **New Dataset** window, you want to set the following fields to the values specified." followed by a list of fields: "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", and "Dataset Name : MNIST Small". At the bottom of the notebook, there is a "New Dataset" button and a "Logout" button.

# 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

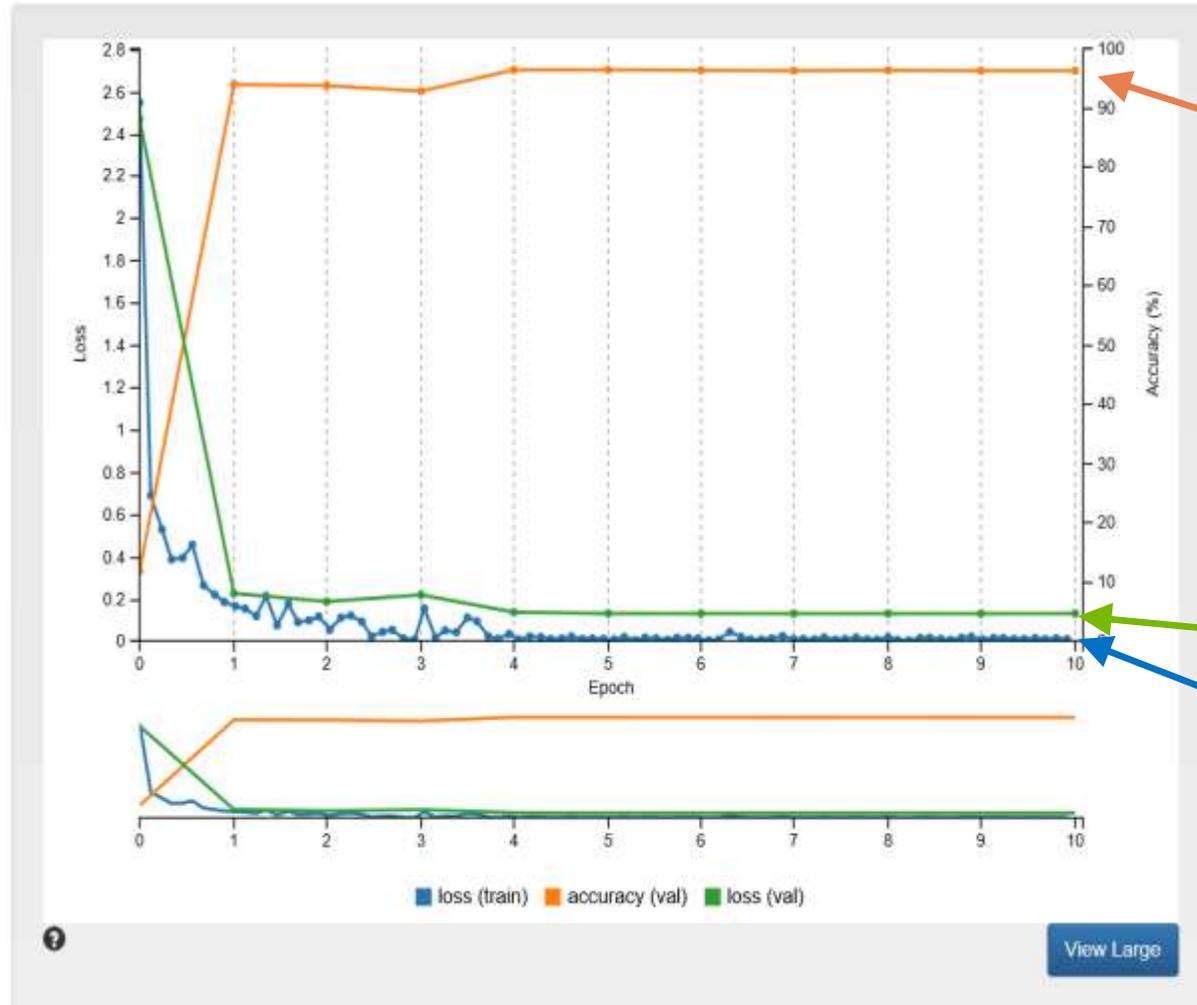
- Dataset settings
  - Image Type: Grayscale
  - Image Size: 28 x 28
  - Training Images: `/home/ubuntu/data/train_small`
  - Select **“Separate test images folder”** checkbox
  - Test Images: `/home/ubuntu/data/test_small`
  - Dataset Name: MNIST Small

# 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

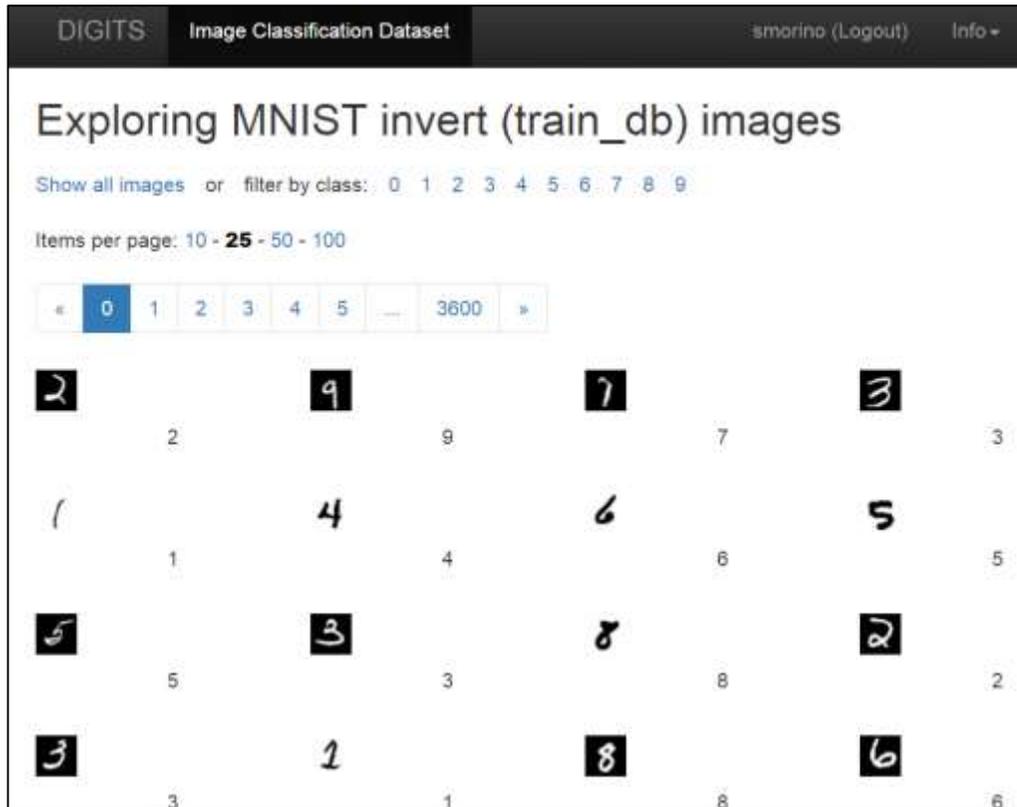
Full dataset ( 10 epochs )

- 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



- $\text{Pixel}(\text{Inverted}) = 255 - \text{Pixel}(\text{original})$
- 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"
}

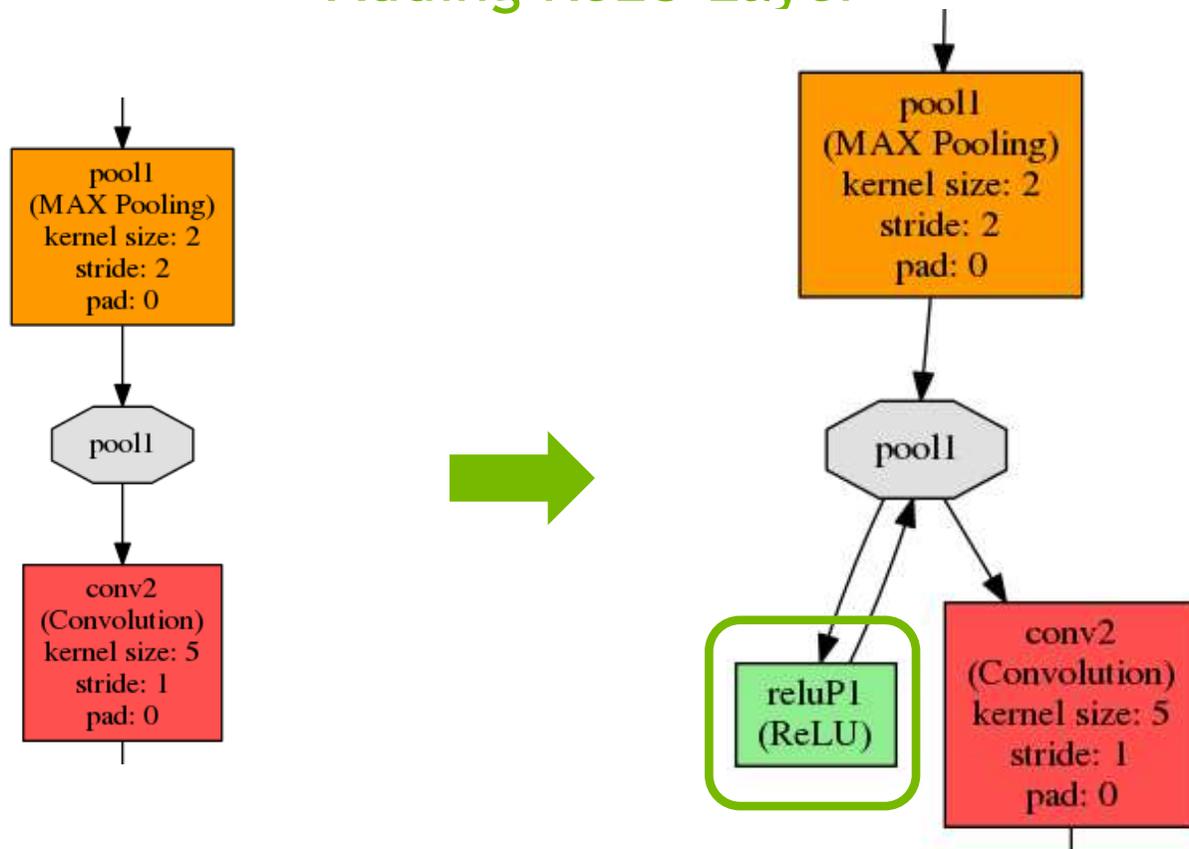
layer {
  name: "reluP1"
```

```
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



DEEP  
LEARNING  
INSTITUTE

[www.nvidia.com/dli](http://www.nvidia.com/dli)