



DEEP
LEARNING
INSTITUTE

Image Classification with DIGITS

Twin Karmakarm

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

TOPICS

- Lab Perspective
- What is Deep Learning
- Handwritten Digit Recognition
- Caffe
- DIGITS
- Lab
 - Discussion / Overview
 - Launching the Lab Environment
 - Lab Review

LAB PERSPECTIVE

WHAT THIS LAB IS

- An introduction to:
 - Deep Learning
 - Workflow of training a network
 - Understanding the results
- Hands-on exercises using Caffe and DIGITS for computer vision and classification

WHAT THIS LAB IS NOT

- Intro to machine learning from first principles
- Rigorous mathematical formalism of neural networks
- Survey of all the features and options of Caffe, DIGITS, or other tools

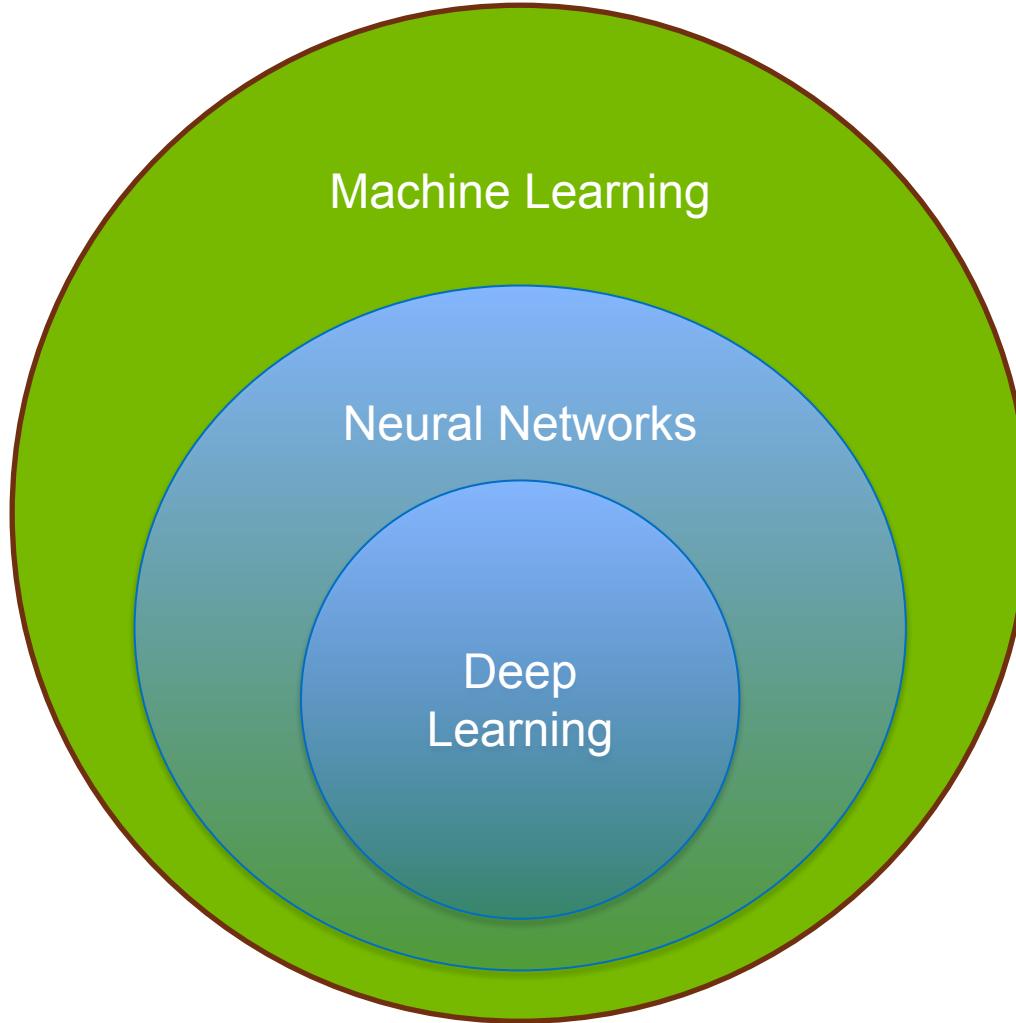
ASSUMPTIONS

- No background in Deep Learning needed
- Understand how to:
 - Navigate a web browser
 - Download files
 - Locate files in file managers

TAKE AWAYS

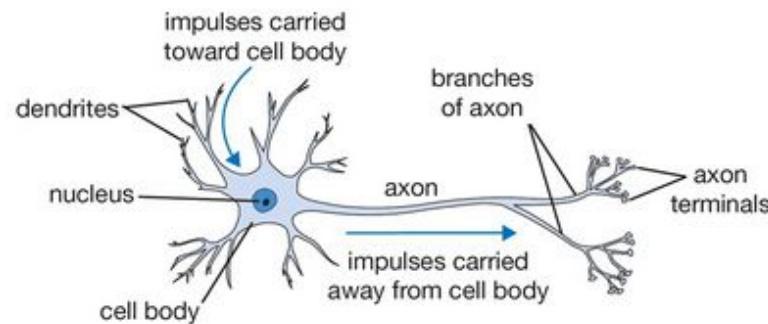
- Understanding of the workflow of Deep Learning
- Ability to setup and train a convolutional neural network
- Enough info to be “dangerous”
 - i.e., you can setup your own CNN and know where to go to learn more

WHAT IS DEEP LEARNING?



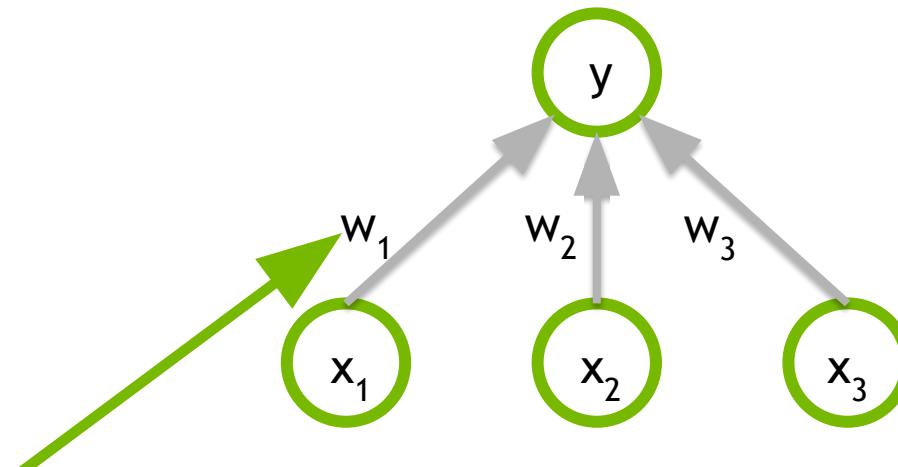
ARTIFICIAL NEURONS

Biological neuron



From Stanford cs231n lecture notes

Artificial neuron

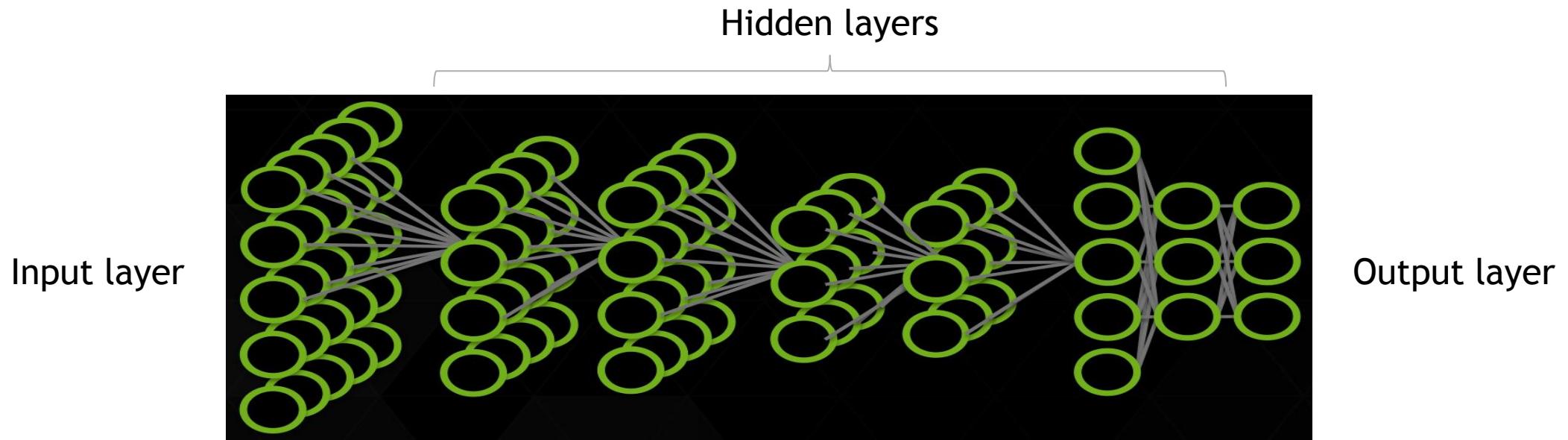


Weights (W_n)
= parameters

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

ARTIFICIAL NEURAL NETWORK

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



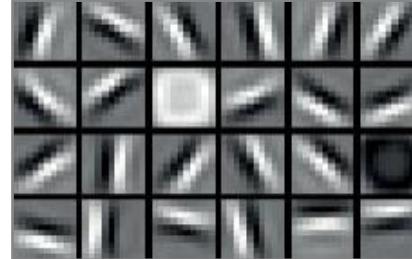
Given sufficient training data an artificial neural network can approximate very complex functions mapping raw data to output decisions

DEEP NEURAL NETWORK (DNN)

Raw data



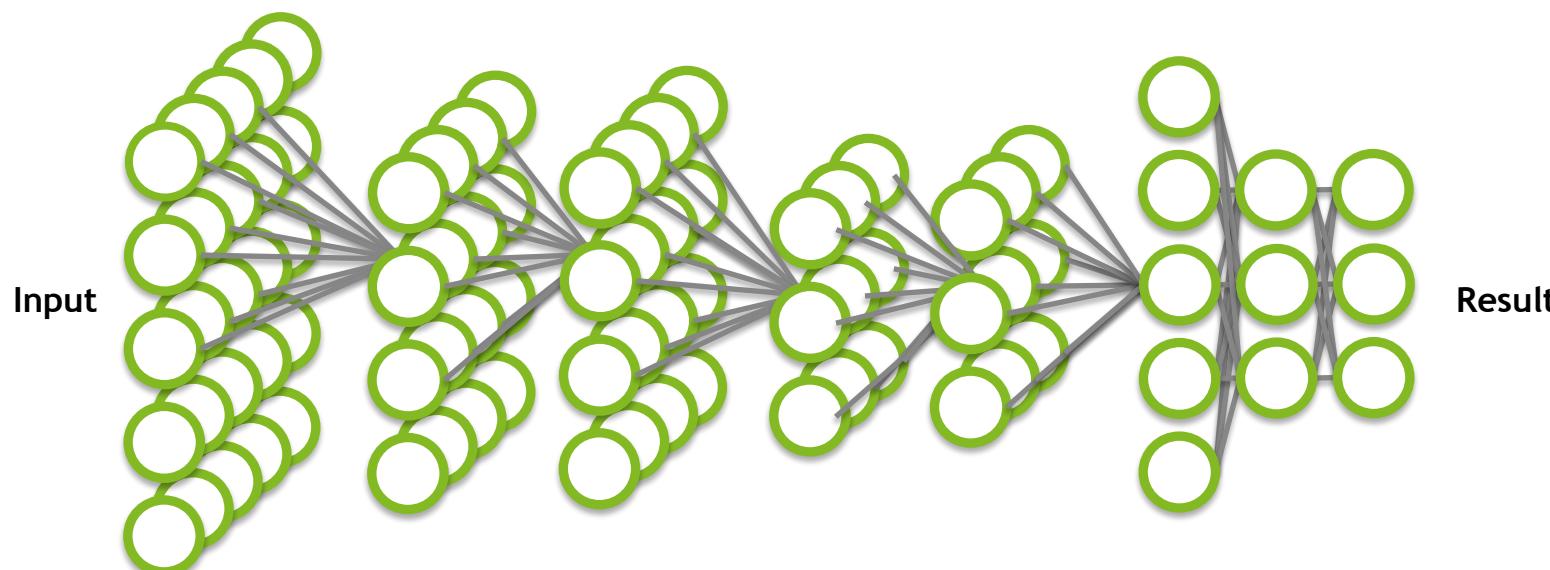
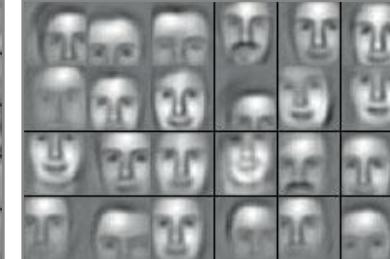
Low-level features



Mid-level features



High-level features



Application components:

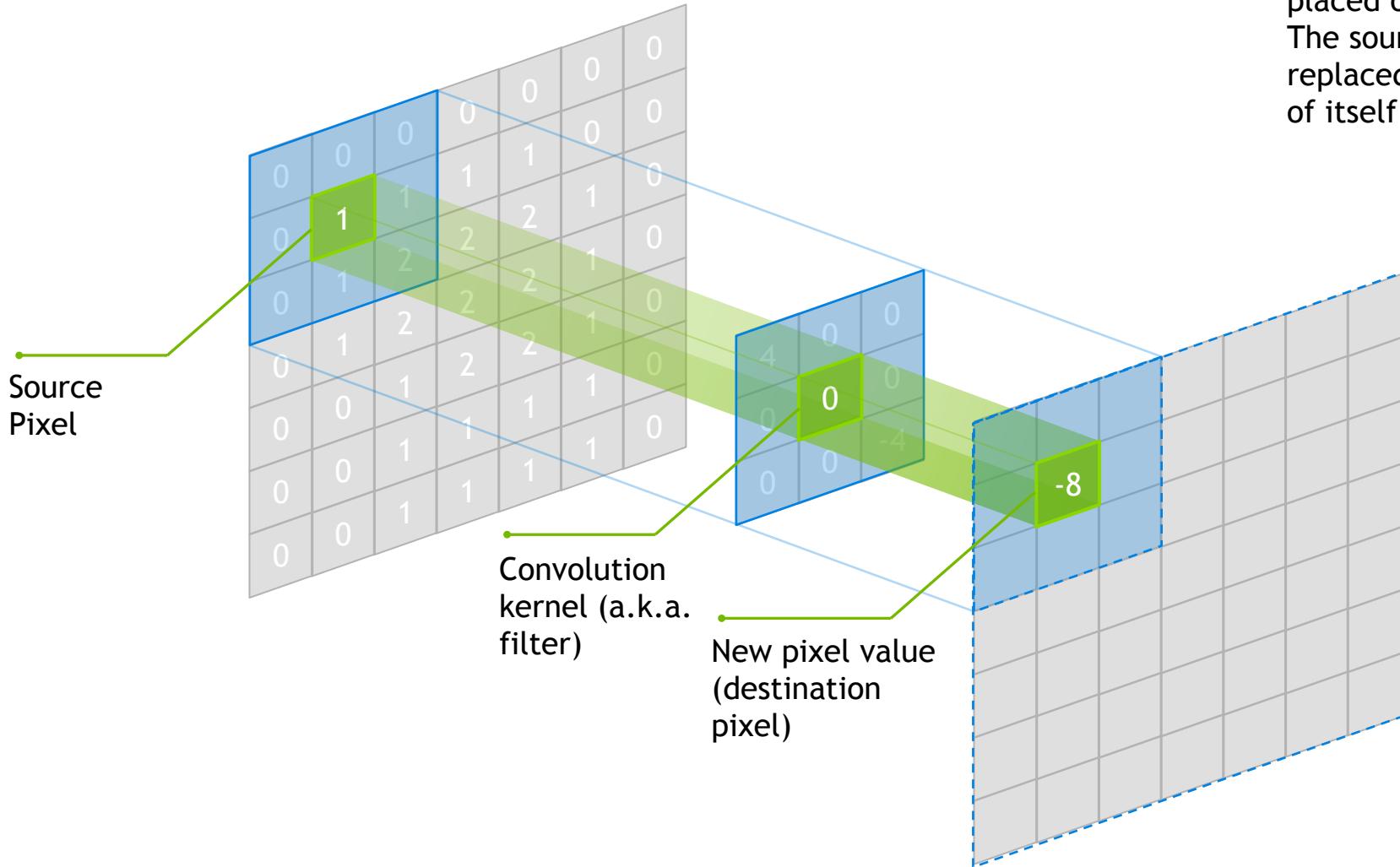
Task objective
e.g. Identify face

Training data
10-100M images

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

Learning algorithm
~30 Exaflops
1-30 GPU days

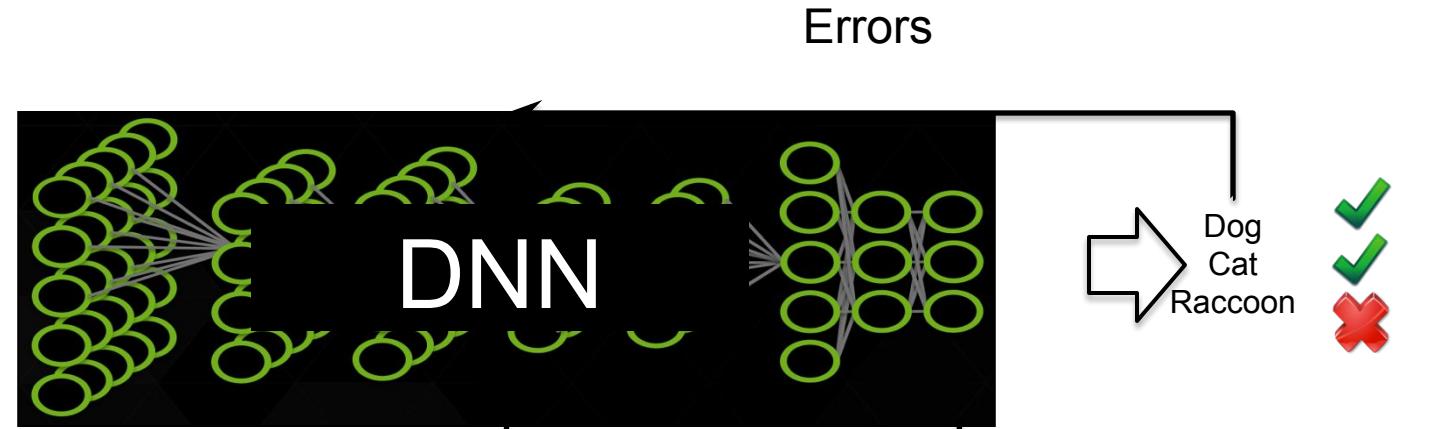
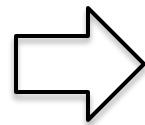
CONVOLUTION



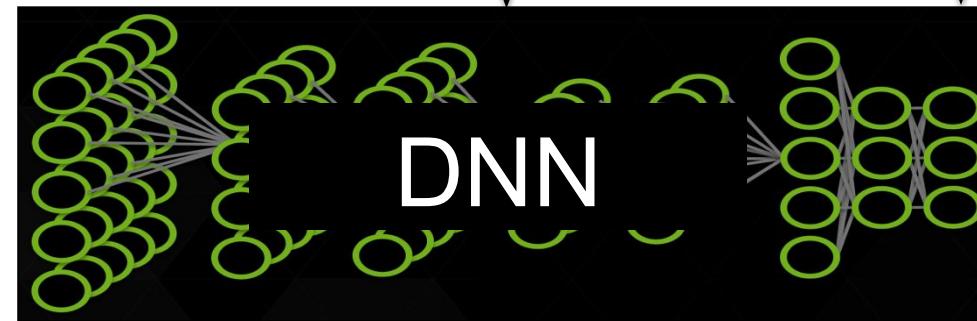
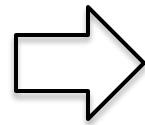
Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

DEEP LEARNING APPROACH

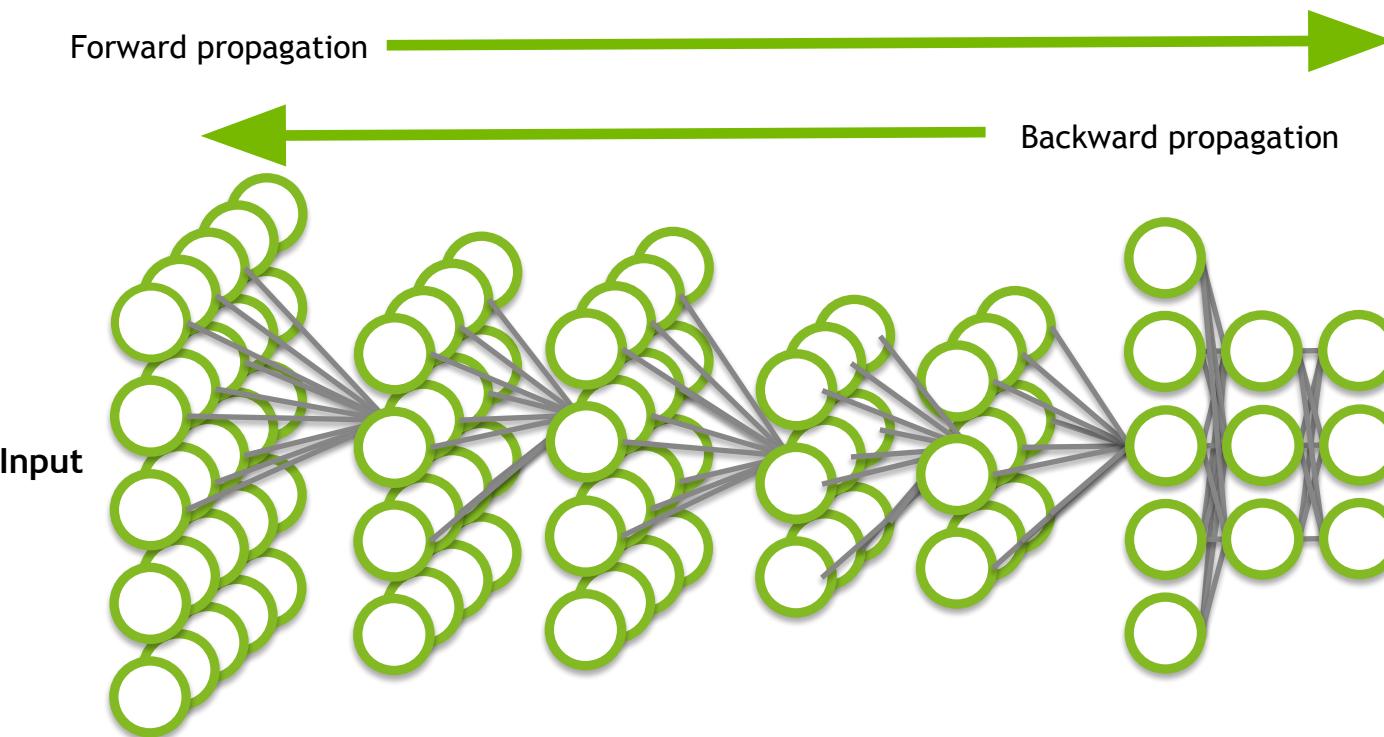
Train:



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

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://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

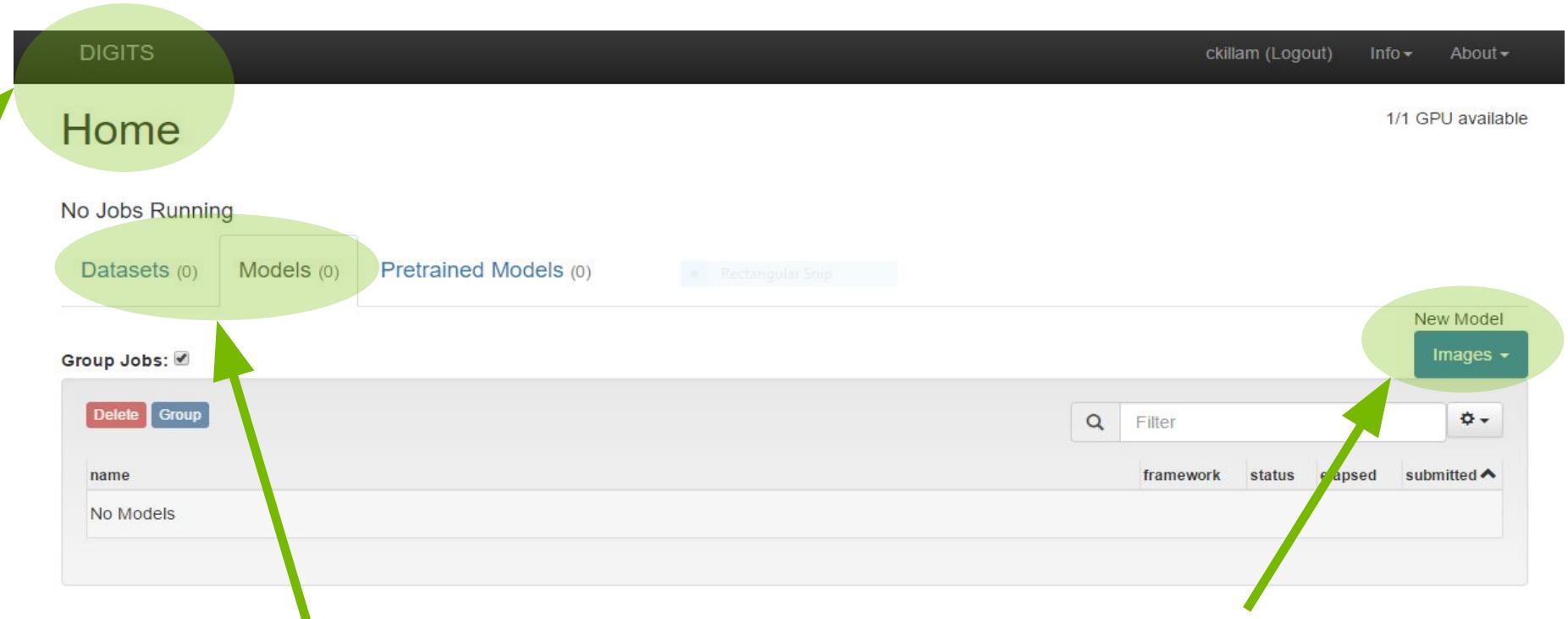
NVIDIA'S DIGITS

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



New Object Detection Dataset

New Image Classification Dataset

Different options will be presented based upon the task

DIGITS - MODEL

New Object Detection Model

Select Dataset

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

Python Layers

- Server-side file: []
- Use client-side file

Network

Details

Intended image size

Standard Networks Previous Networks Pretrained Networks Custom Network

New Image Classification Model

Select Dataset

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

Python Layers

- Server-side file: []
- Use client-side file

Network

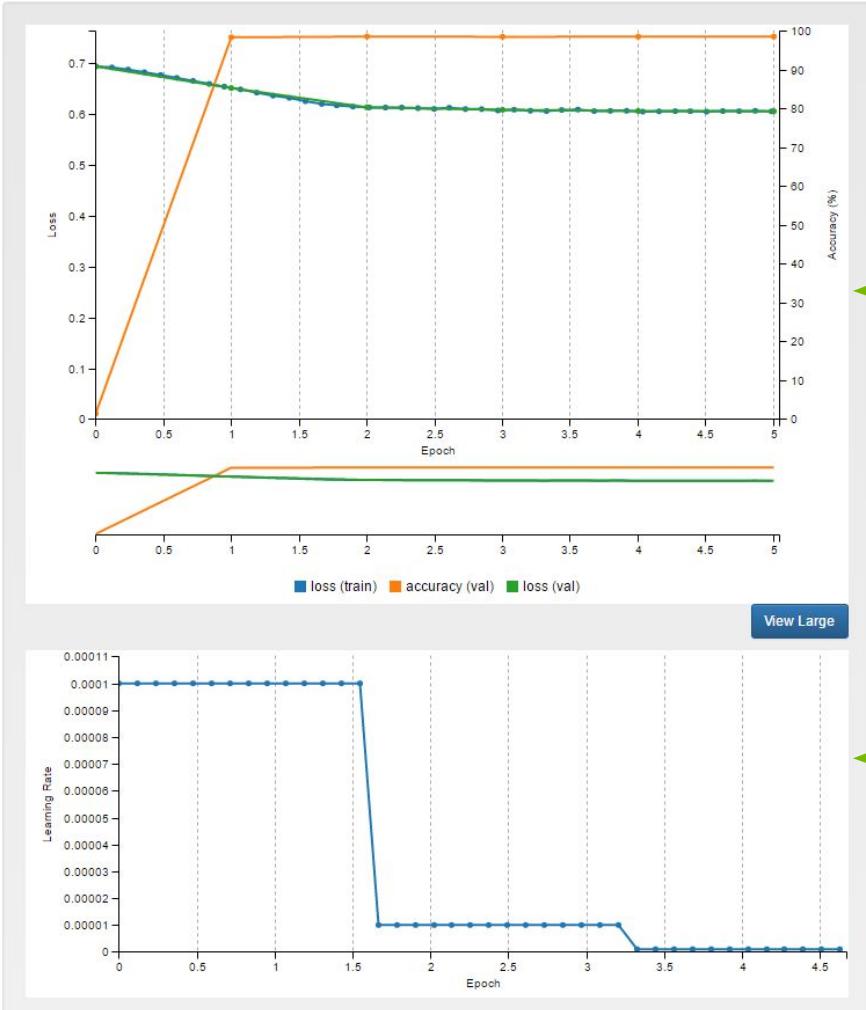
Standard Networks	Previous Networks	Pretrained Networks	Custom Network
Caffe	Torch		
LeNet	Original paper [1998]		28x28 (gray)

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

Trained Models

Select Model

Epoch #5

Download Model

Make Pretrained Model

Select Visualization Method

Image Segmentation

Visualization Options

Display segmented image.

Colormap ?

From dataset

Inference Options

Do not resize input image(s) ?

Test an image

Image file ?

image file

Test a record from validation set

Record from validation set ?

SC-HF-NI-3

Show visualizations and statistics ?

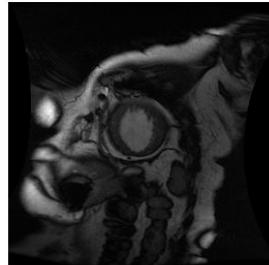
Test

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

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	

LAB DISCUSSION / OVERVIEW

LAB OVERVIEW

- Learn about the workflow of Deep Learning
 - Create dataset
 - Create model
 - Evaluate model results
 - Try different techniques to improve initial results
- Train your own Convolutional Neural Network using Caffe and DIGITS to identify handwritten characters

CREATE DATASET IN DIGITS

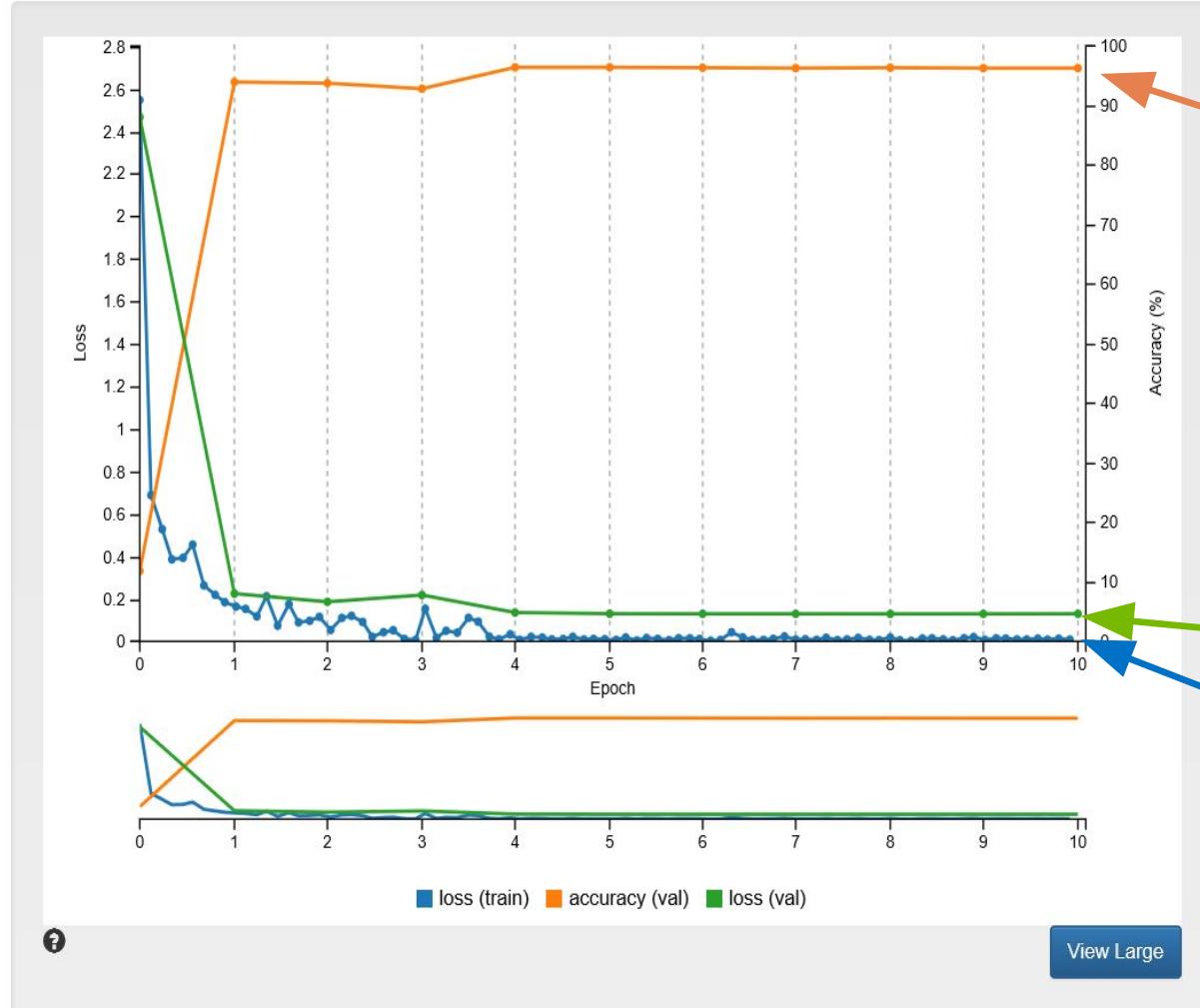
- Dataset settings
 - Image Type: Grayscale
 - Image Size: 28 x 28
 - Training Images: /home/ubuntu/data/train_small
 - Select “Separate test images folder” checkbox
 - Test Images: /home/ubuntu/data/test_small
 - Dataset Name: MNIST Small

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)

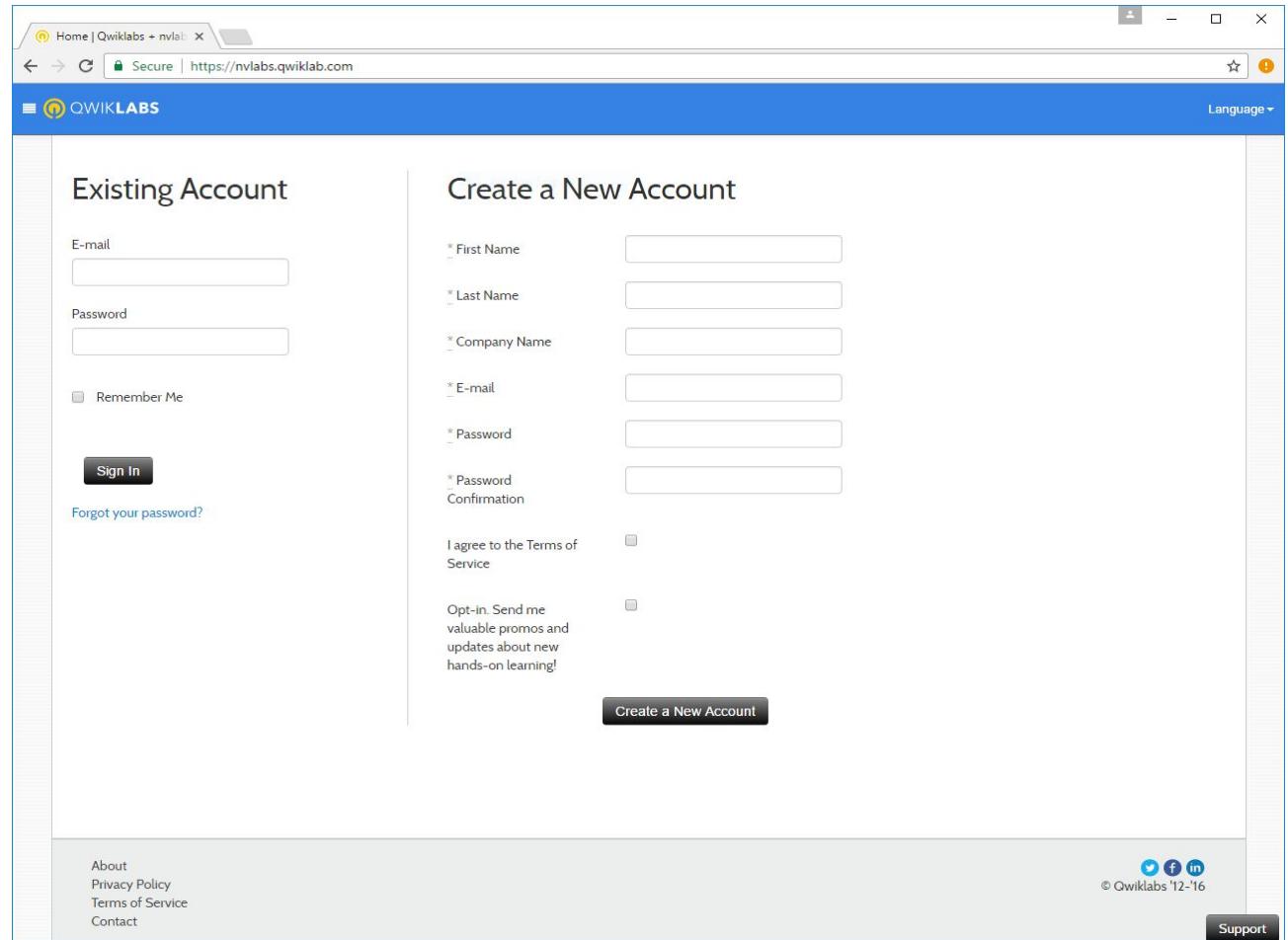
ADDITIONAL TECHNIQUES TO IMPROVE MODEL

- More training data
- Data augmentation
- Modify the network

LAUNCHING THE LAB ENVIRONMENT

NAVIGATING TO QWIKLABS

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



The screenshot shows a web browser window for the QwikLabs website. The URL in the address bar is <https://nvlabs.qwiklab.com>. The page has a blue header with the QwikLabs logo. Below the header, there are two main sections: "Existing Account" on the left and "Create a New Account" on the right.

Existing Account: This section contains fields for "E-mail" and "Password", and a "Remember Me" checkbox. A "Sign In" button is located below these fields. A link "Forgot your password?" is also present.

Create a New Account: This section contains fields for "First Name", "Last Name", "Company Name", "E-mail", "Password", and "Password Confirmation". There is a checkbox for "I agree to the Terms of Service" and another for "Opt-in. Send me valuable promos and updates about new hands-on learning!". A "Create a New Account" button is at the bottom of this section.

Footer: The footer includes links for "About", "Privacy Policy", "Terms of Service", and "Contact". It also features social media icons for Twitter, Facebook, and LinkedIn, and a "Support" button.

ACCESSING LAB ENVIRONMENT

3. Select the event specific In-Session Class in the upper left
4. Click the “Image Classification with DIGITS” Class from the list

The screenshot shows the NVIDIA Deep Learning Institute Lab Environment. At the top, there is a header bar with the text "In-Session Class: GTC2017" in a dropdown menu, followed by a clock icon, "125.3 Total Hours", a green circular icon with three dots labeled "68 Completed Labs", and a blue circular icon with three horizontal bars labeled "8 Classes Taken". Below the header is a section titled "Class Details" which lists several lab classes:

- Deep Learning for Image Segmentation
- Neural Network Deployment with DIGITS and TensorRT
- Image Classification with DIGITS** (This class is highlighted with a green background)
- 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++

To the right of the list, there is a detailed description of the selected class, "Image Classification with DIGITS":

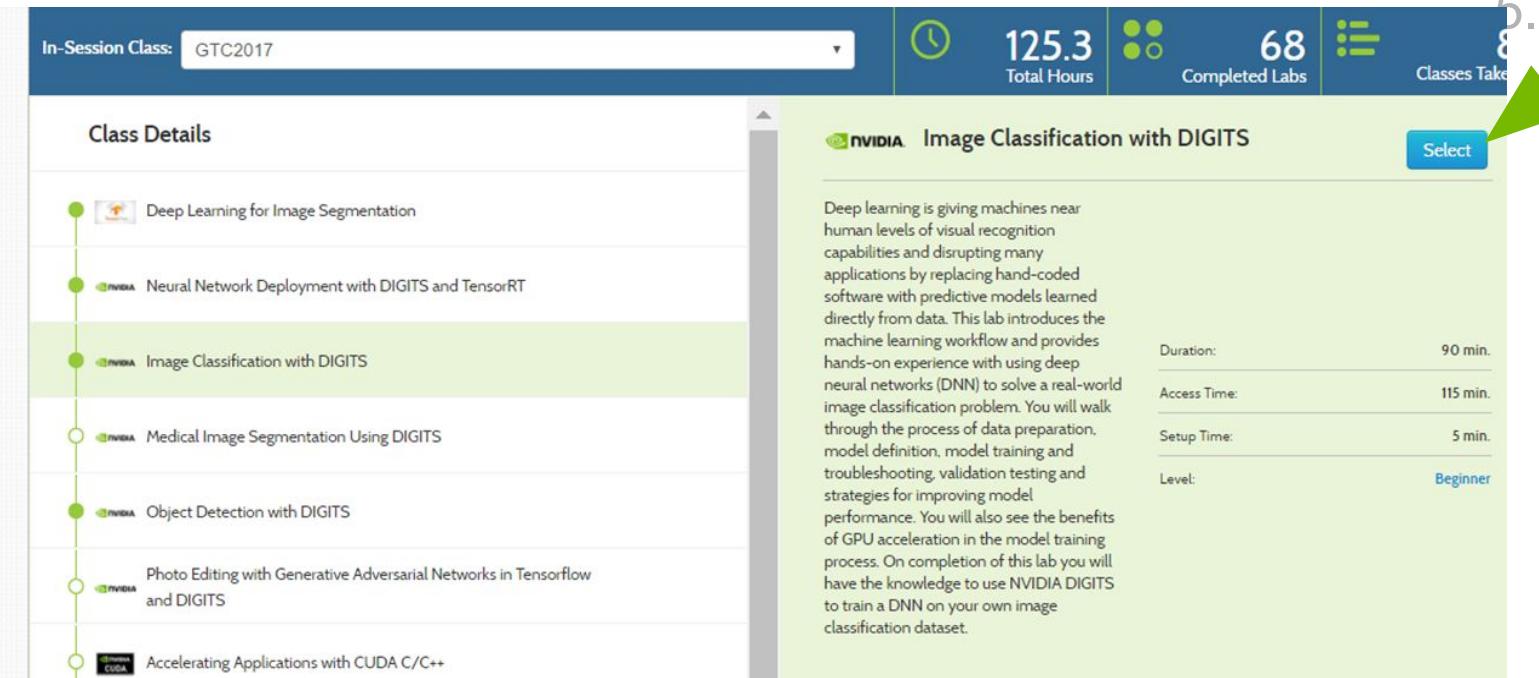
nVIDIA Image Classification with DIGITS

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

A blue "Select" button is located at the top right of the class details panel.

LAUNCHING THE LAB ENVIRONMENT



The screenshot shows the NVIDIA Deep Learning Institute dashboard. At the top, there's a header with 'In-Session Class: GTC2017', a clock icon, '125.3 Total Hours', a completed labs icon with '68 Completed Labs', and a 'Classes Taken' icon with '5.8'. Below the header, there's a sidebar titled 'Class Details' listing several labs:

- Deep Learning for Image Segmentation
- Neural Network Deployment with DIGITS and TensorRT
- Image Classification with DIGITS** (highlighted in green)
- 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++

The main content area displays details for the selected lab, 'Image Classification with DIGITS':

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

A blue 'Select' button is visible at the top right of the lab card. A green arrow points from the number '5.8' in the header to this 'Select' button.

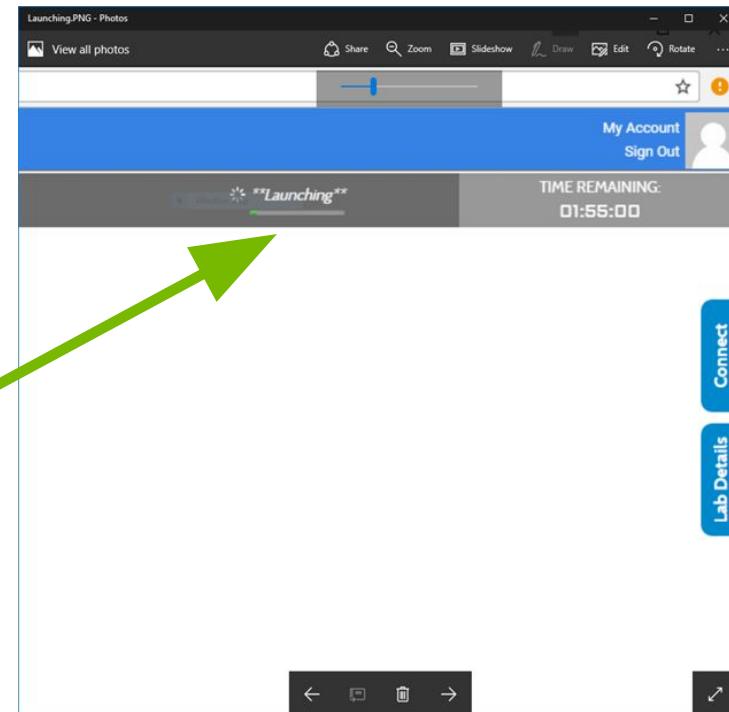
Click on the Select button to launch the lab environment

- 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

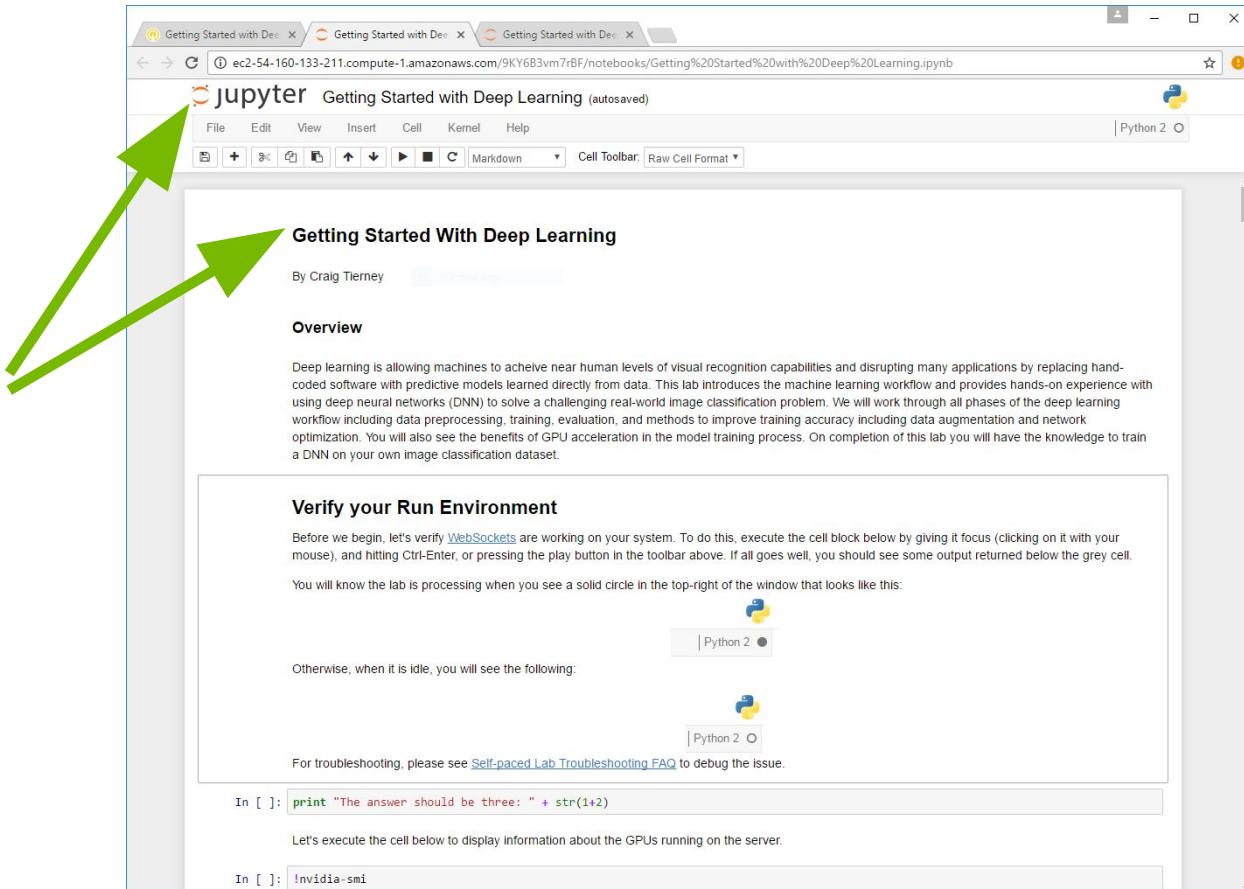
CONNECTING TO THE LAB ENVIRONMENT



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

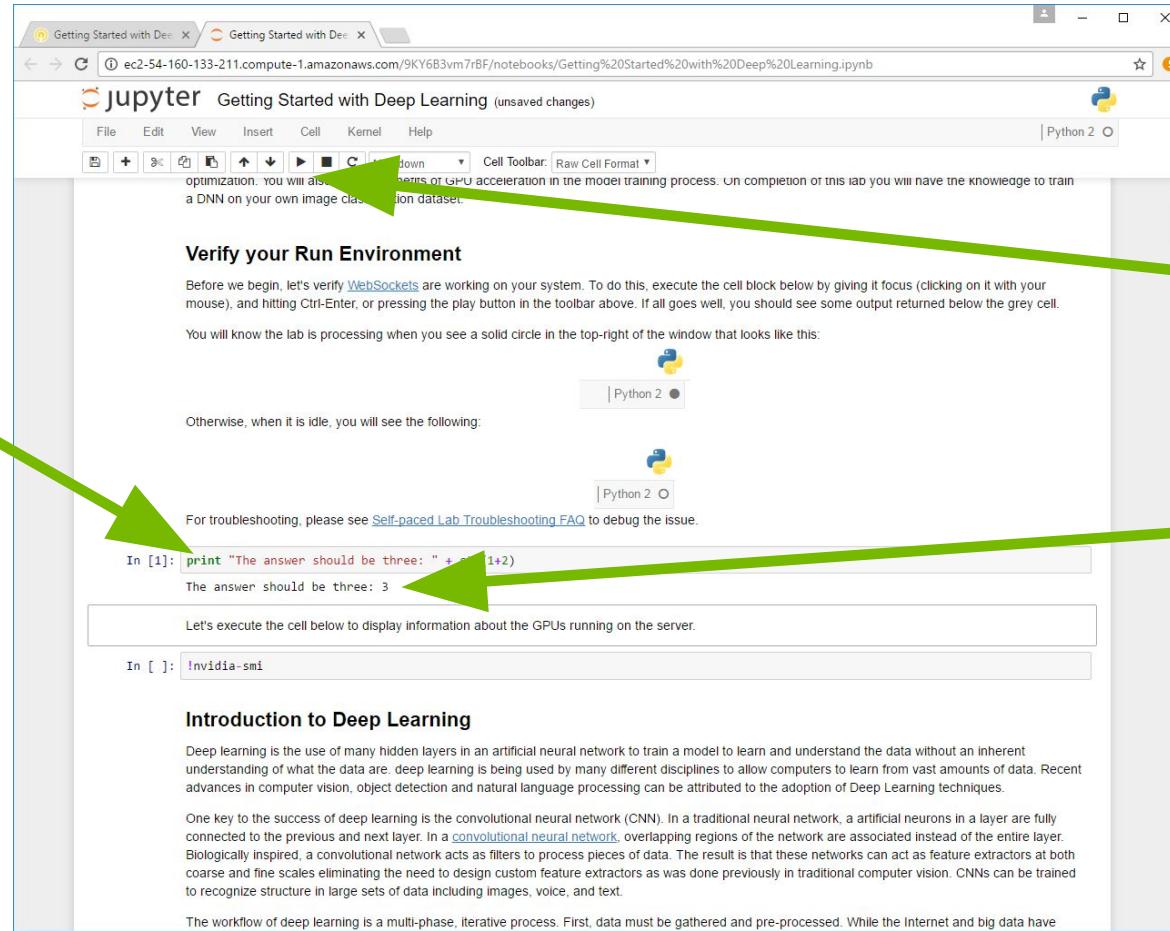
CONNECTING TO THE LAB ENVIRONMENT

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



JUPYTER NOTEBOOK

1. Place your cursor in the code

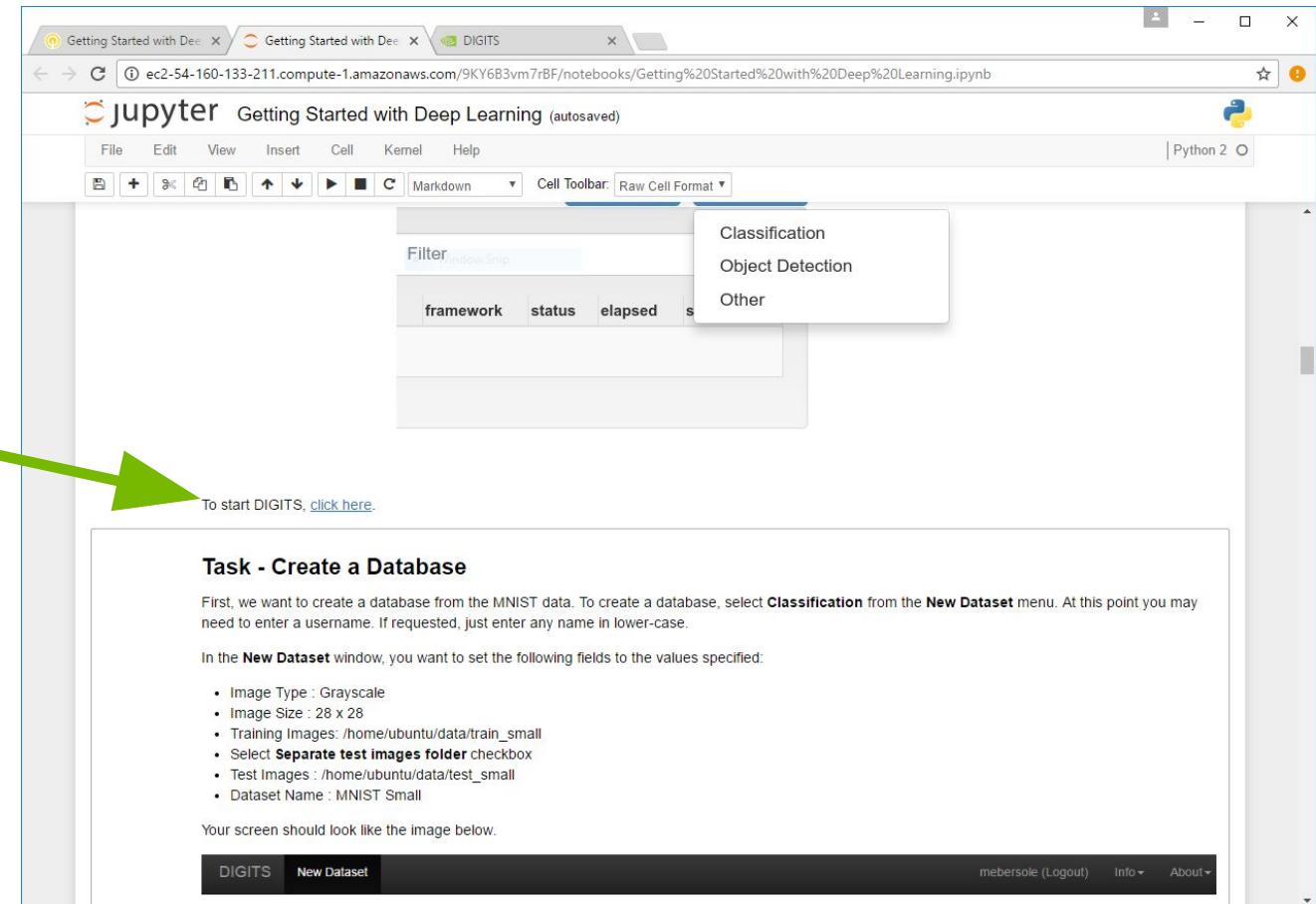


2. Click the “run cell” button

3. Confirm you receive the same result

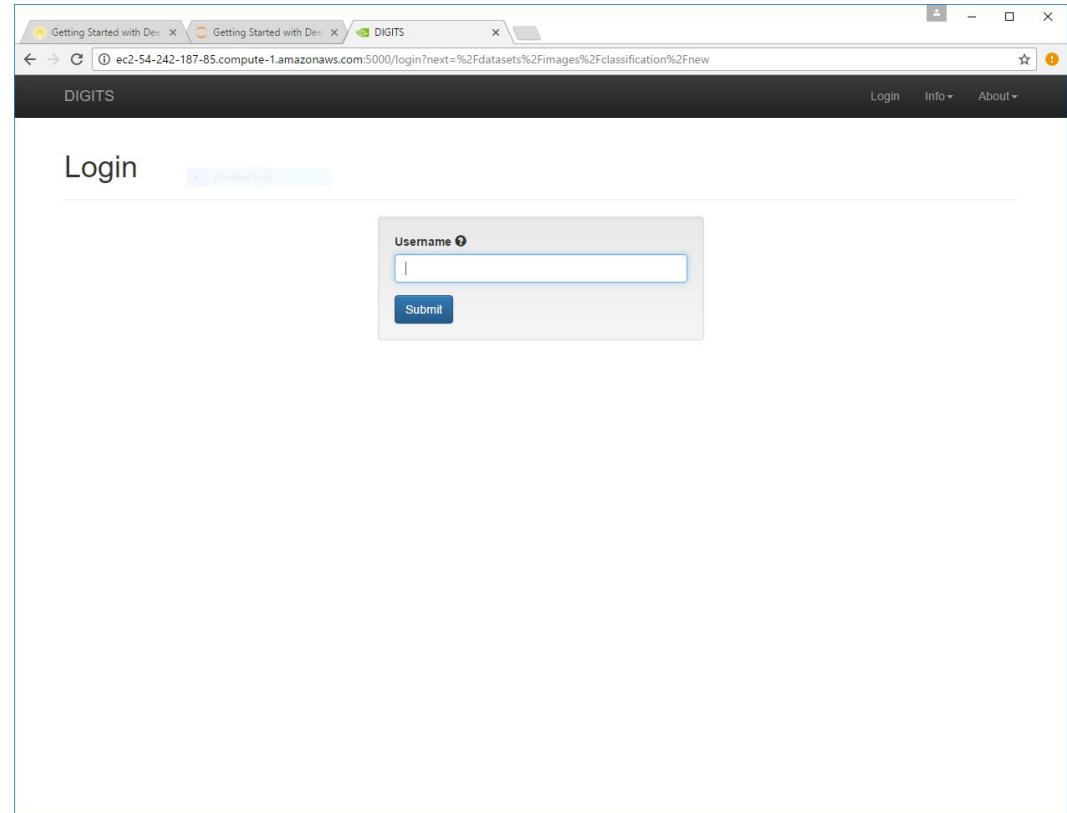
STARTING DIGITS

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 REVIEW

FIRST RESULTS

Small dataset (10 epochs)

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

SMALL DATASET	
1	1 : 99.90 %
2	2 : 69.03 %
3	8 : 71.37 %
4	8 : 85.07 %
7	0 : 99.00 %
8	8 : 99.69 %
8	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	1 : 99.90 %	0 : 93.11 %
2	2 : 69.03 %	2 : 87.23 %
3	8 : 71.37 %	8 : 71.60 %
4	8 : 85.07 %	8 : 79.72 %
7	0 : 99.00 %	0 : 95.82 %
8	8 : 99.69 %	8 : 100.0 %
8	8 : 54.75 %	2 : 70.57 %

DATA AUGMENTATION

Adding Inverted Images

DIGITS Image Classification Dataset smorino (Logout) Info ▾

Exploring MNIST invert (train_db) images

Show all images or filter by class: 0 1 2 3 4 5 6 7 8 9

Items per page: 10 - 25 - 50 - 100

« 0 1 2 3 4 5 ... 3600 »

2 9 7 3
2 9 7 3
1 4 6 5
1 4 6 5
5 3 8 2
5 3 8 2
3 1 8 6
3 1 8 6

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

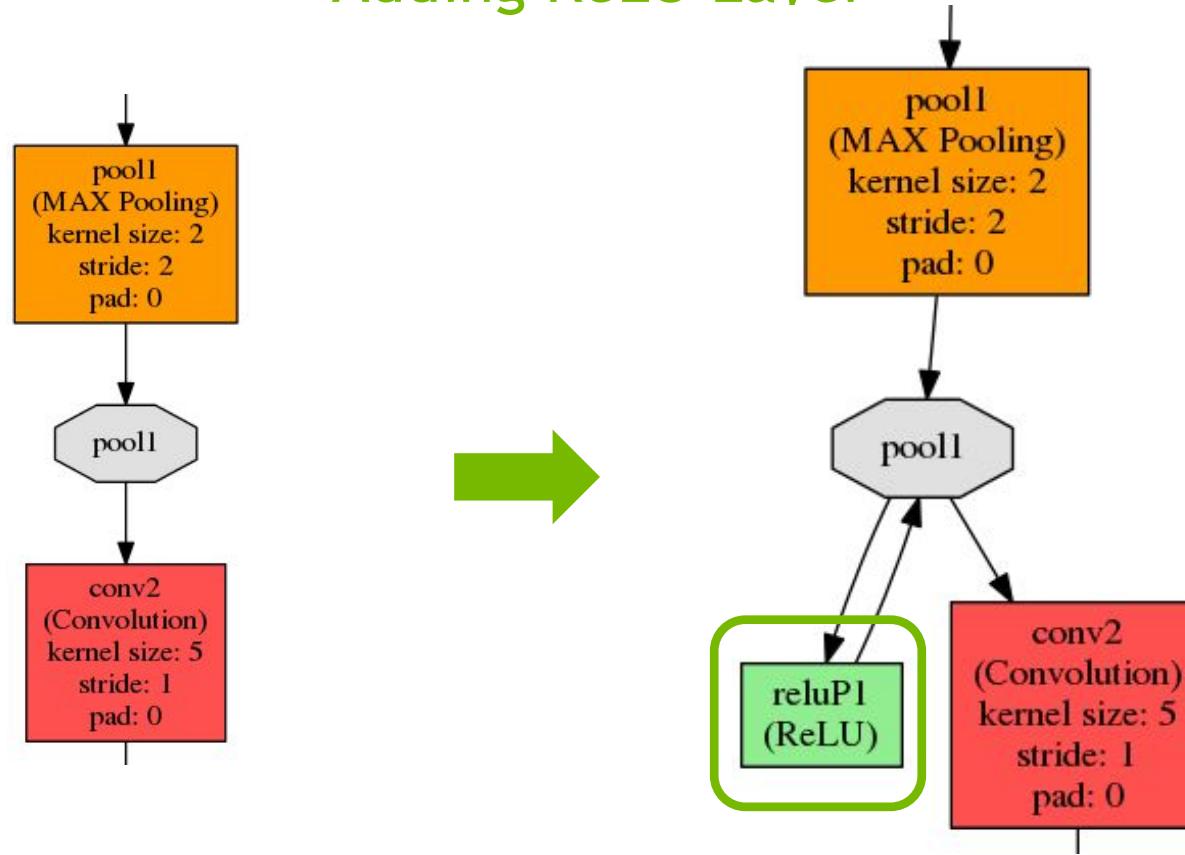
Adding filters and ReLU layer

```
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	1 : 99.90 %	0 : 93.11 %	1 : 90.84 %	1 : 59.18 %
2	2 : 69.03 %	2 : 87.23 %	2 : 89.44 %	2 : 93.39 %
3	8 : 71.37 %	8 : 71.60 %	3 : 100.0 %	3 : 100.0 %
4	8 : 85.07 %	8 : 79.72 %	4 : 100.0 %	4 : 100.0 %
7	0 : 99.00 %	0 : 95.82 %	7 : 82.84 %	2 : 62.52 %
8	8 : 99.69 %	8 : 100.0 %	8 : 100.0 %	8 : 100.0 %
8	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

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

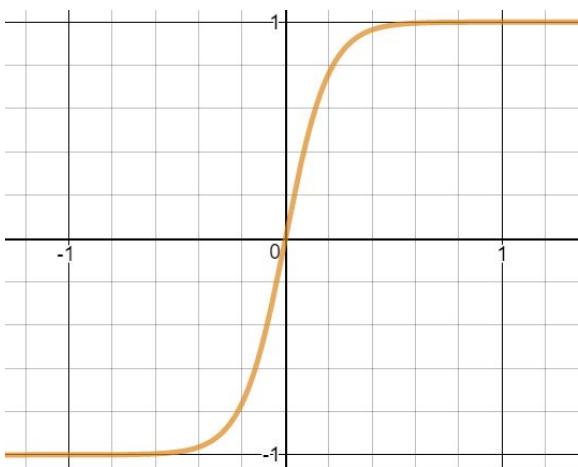


www.nvidia.com/dli

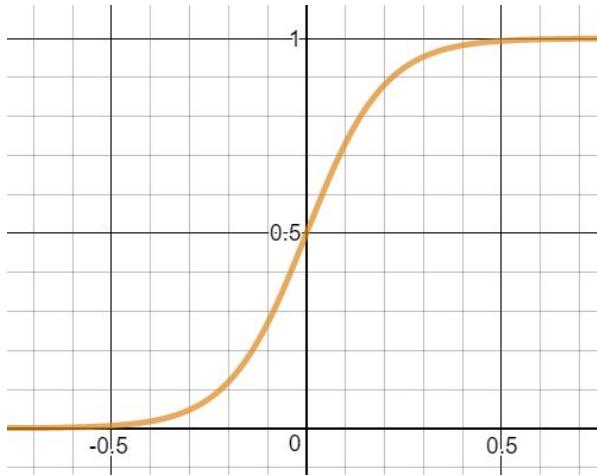
DEEP
LEARNING
INSTITUTE

Instructor: Charles Killam, LP.D.

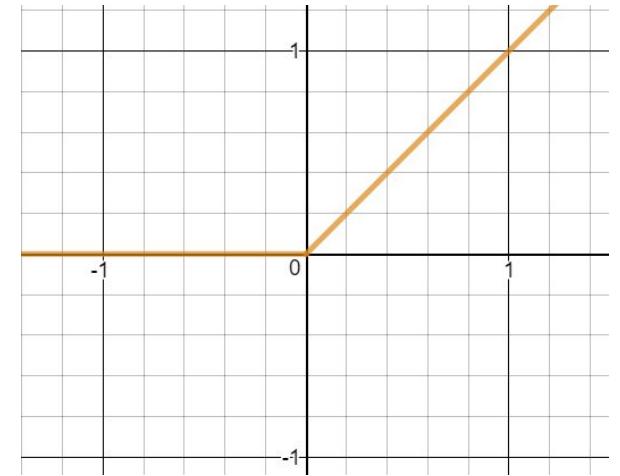
Activation functions



tanh



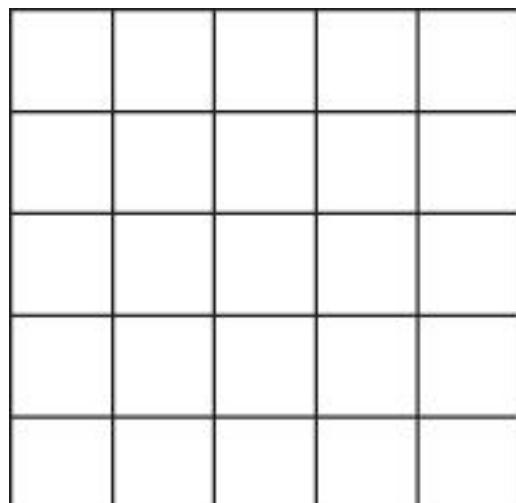
Sigmoid



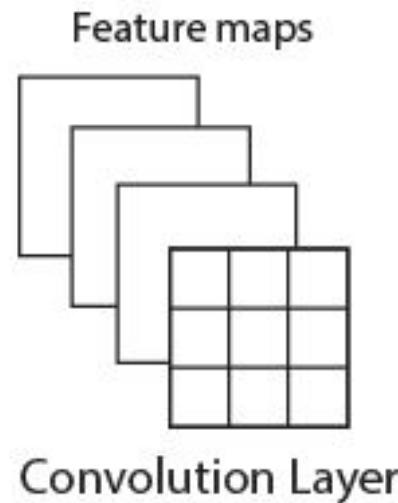
ReLU

CNN - Example

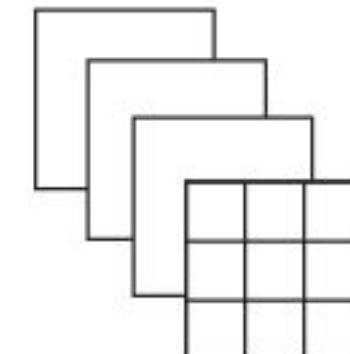
- Each pixel is a neuron



Input image



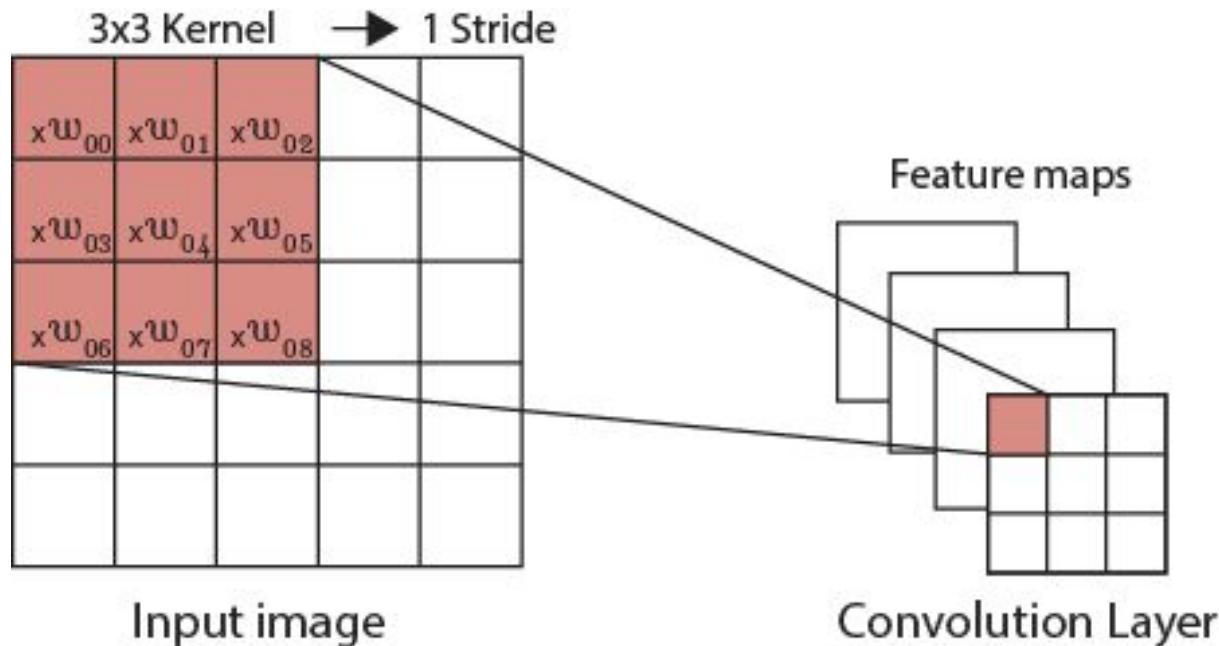
Feature maps



Convolution Layer

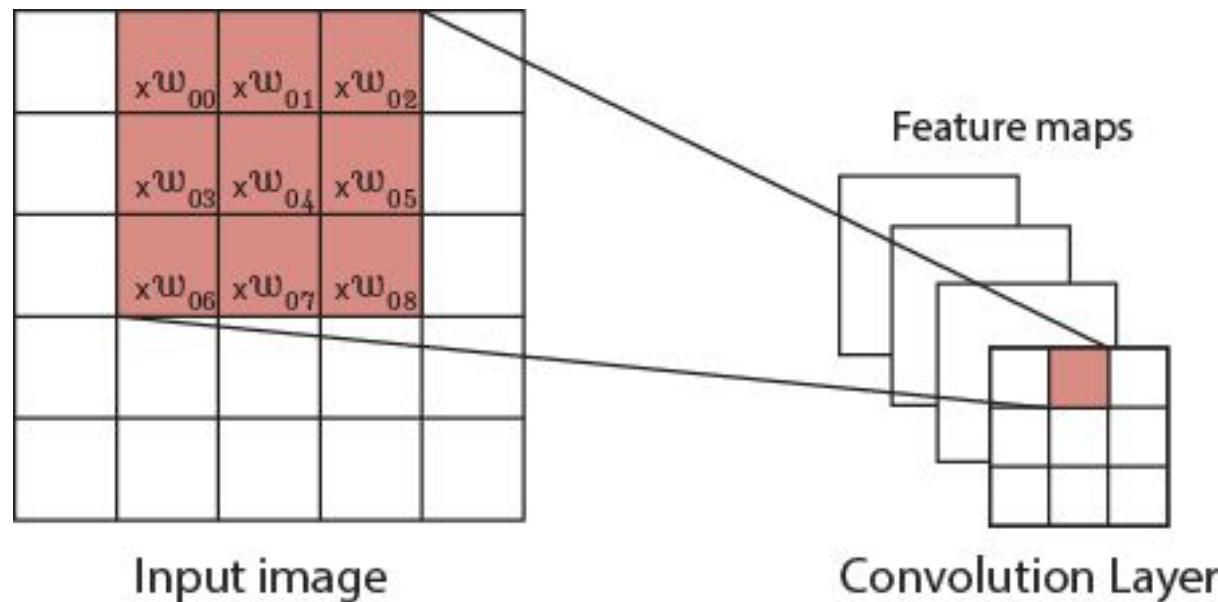
CNN - Example - 1st Feature Map

- 3x3 Kernel, 1 Stride, weights constant per kernel



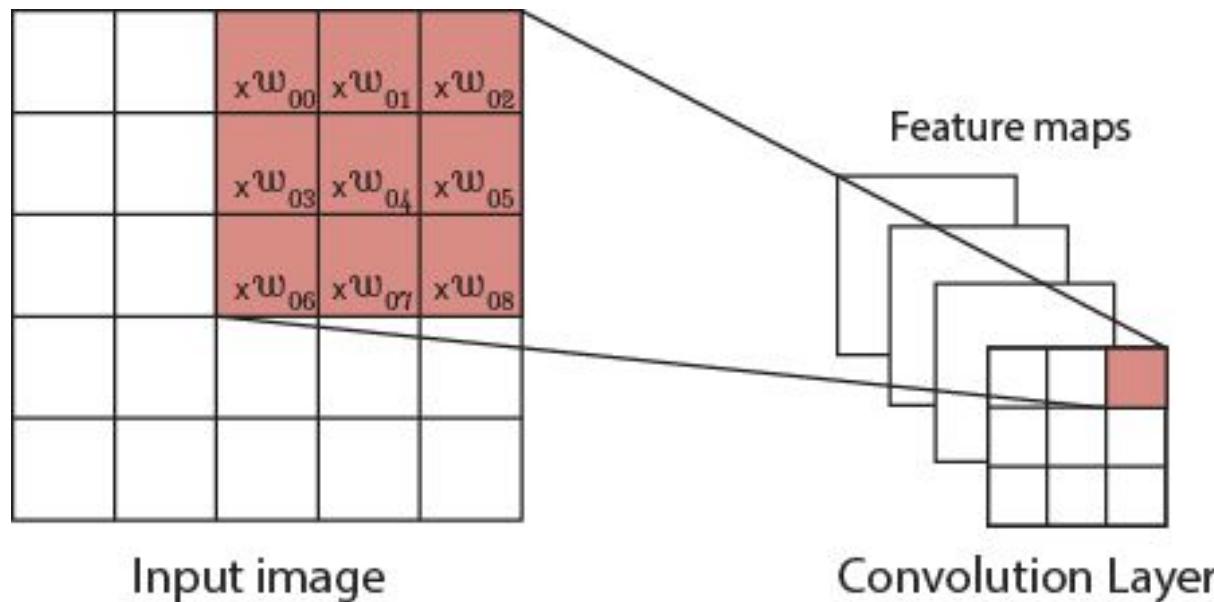
CNN - Example - 1st Feature Map

- 3x3 Kernel, 1 Stride, weights constant per kernel



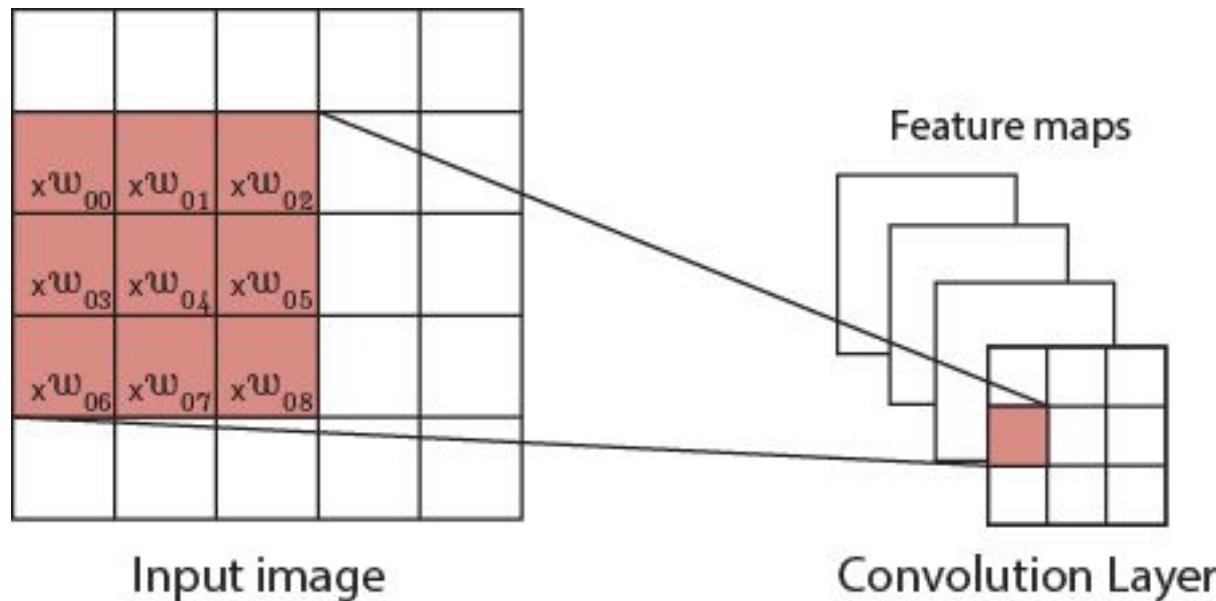
CNN - Example - 1st Feature Map

- 3x3 Kernel, 1 Stride, weights constant per kernel



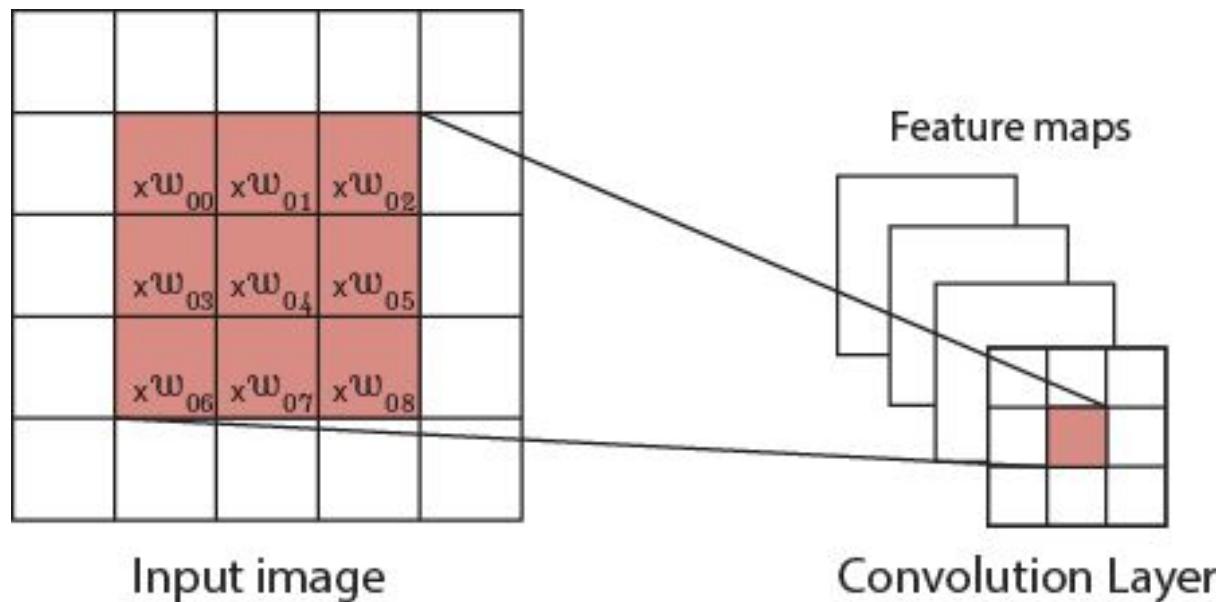
CNN - Example - 1st Feature Map

- 3x3 Kernel, 1 Stride, weights constant per kernel



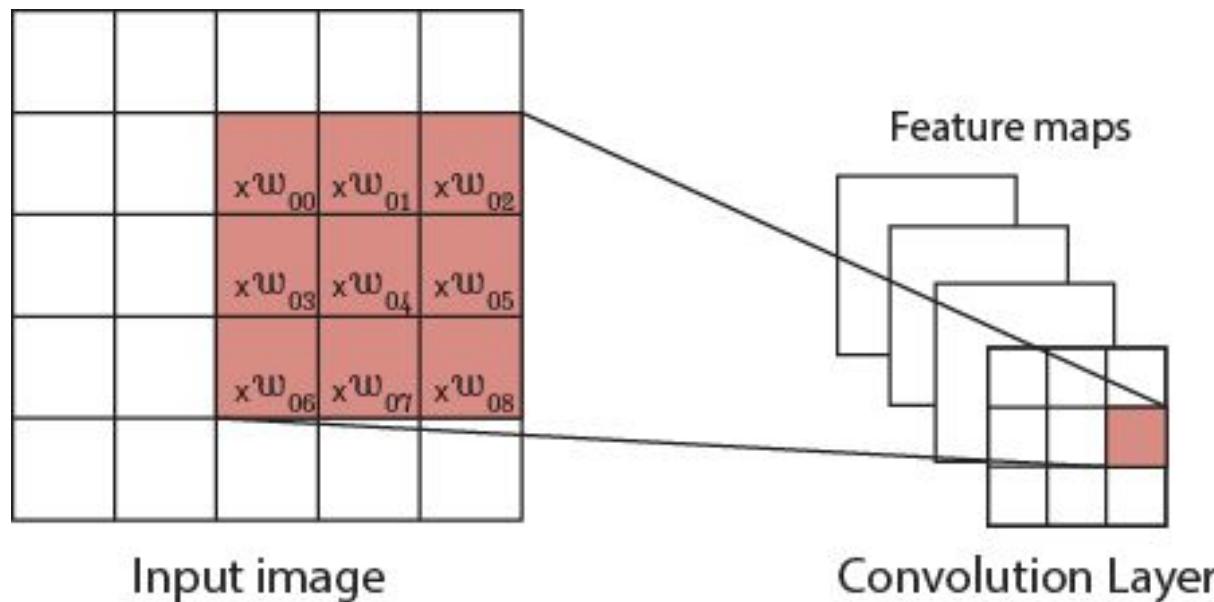
CNN - Example - 1st Feature Map

- 3x3 Kernel, 1 Stride, weights constant per kernel

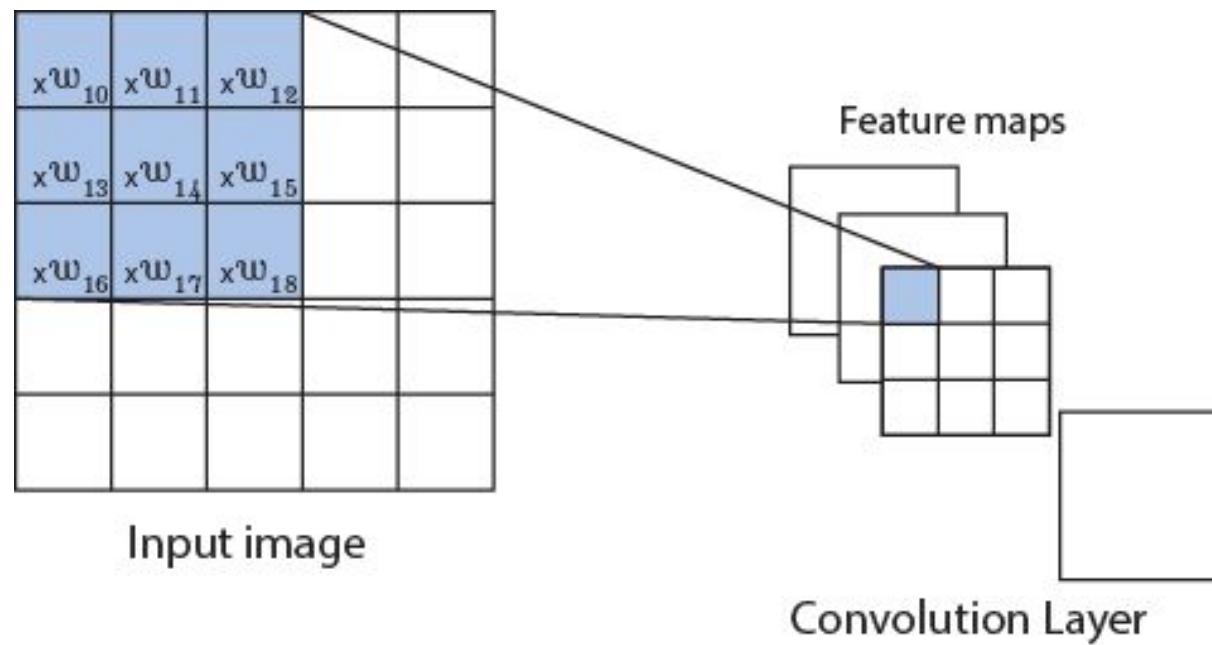


CNN - Example - 1st Feature Map

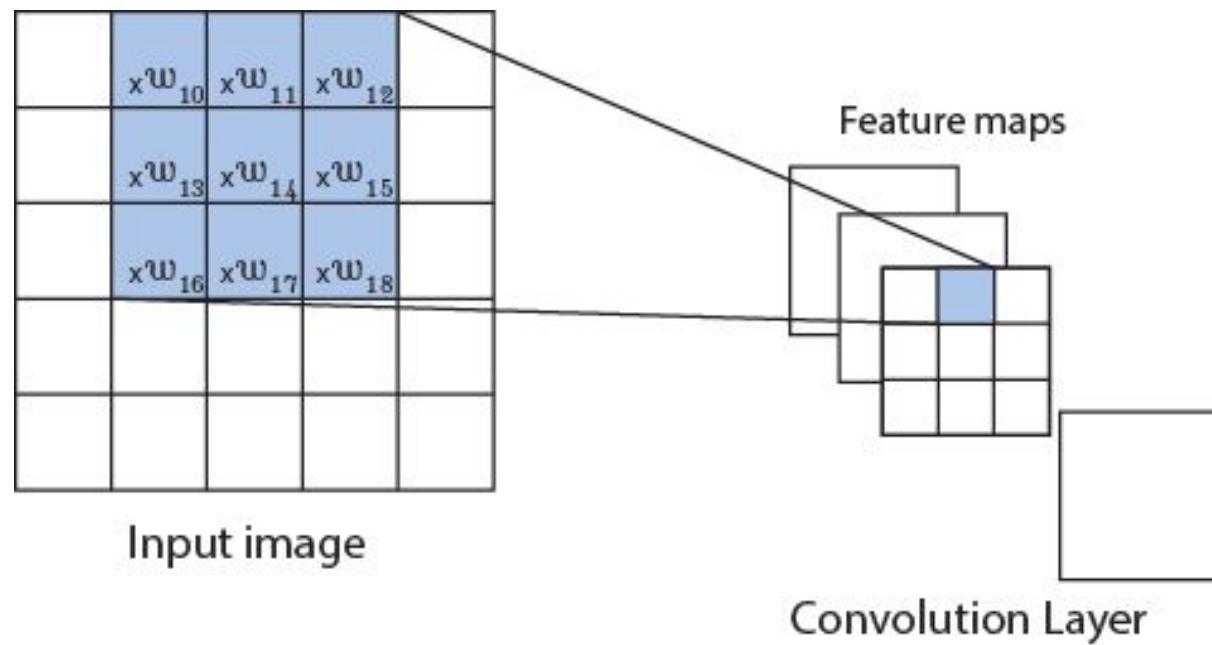
- 3x3 Kernel, 1 Stride, weights constant per kernel



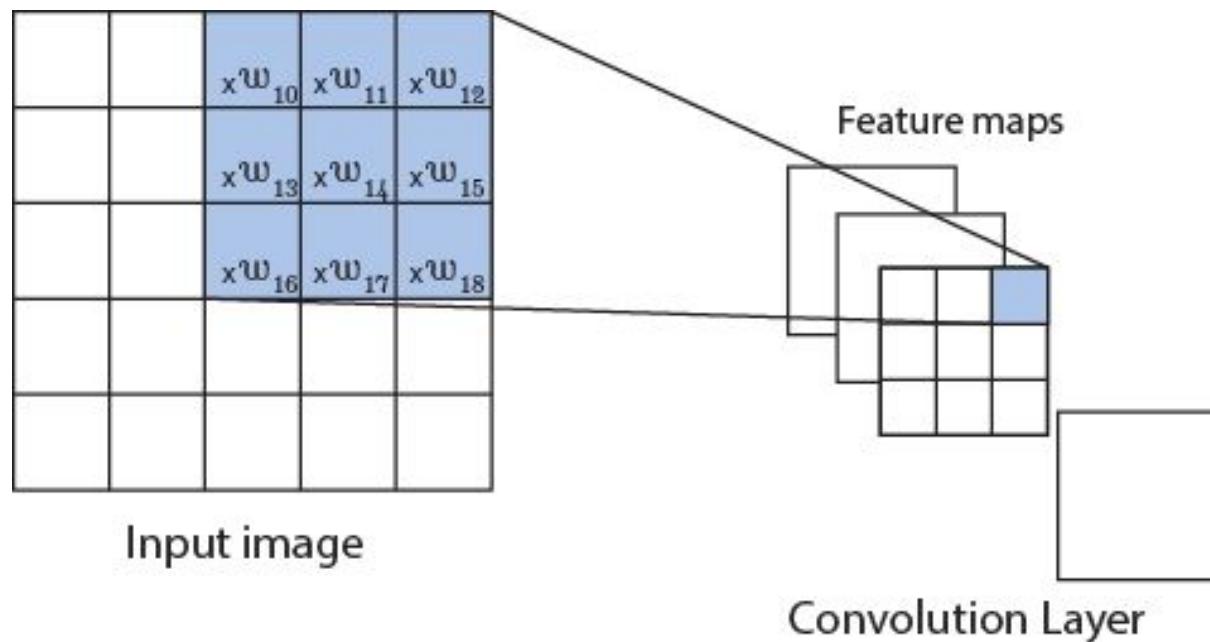
CNN - Example - 2nd Feature Map



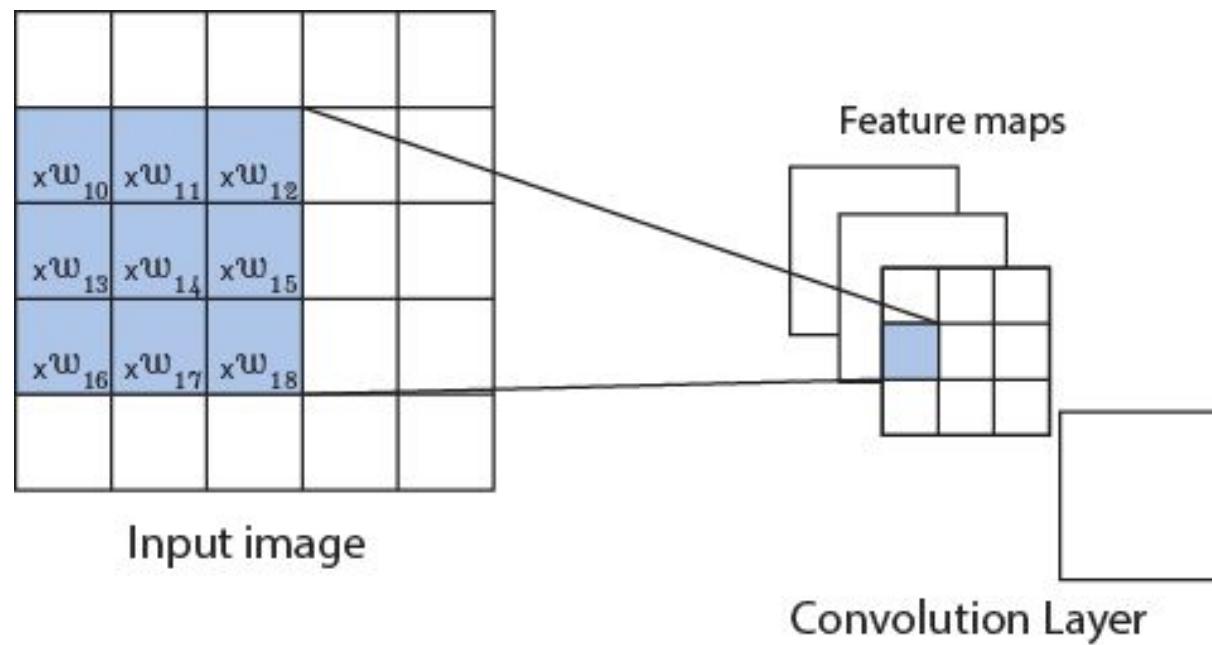
CNN - Example - 2nd Feature Map



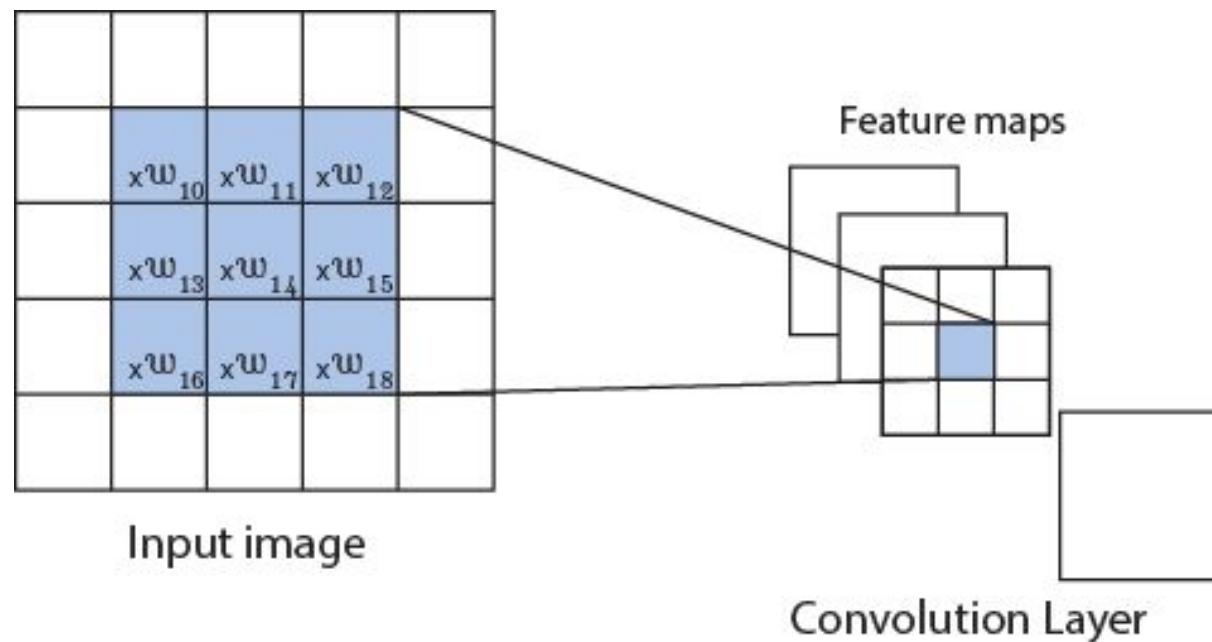
CNN - Example - 2nd Feature Map



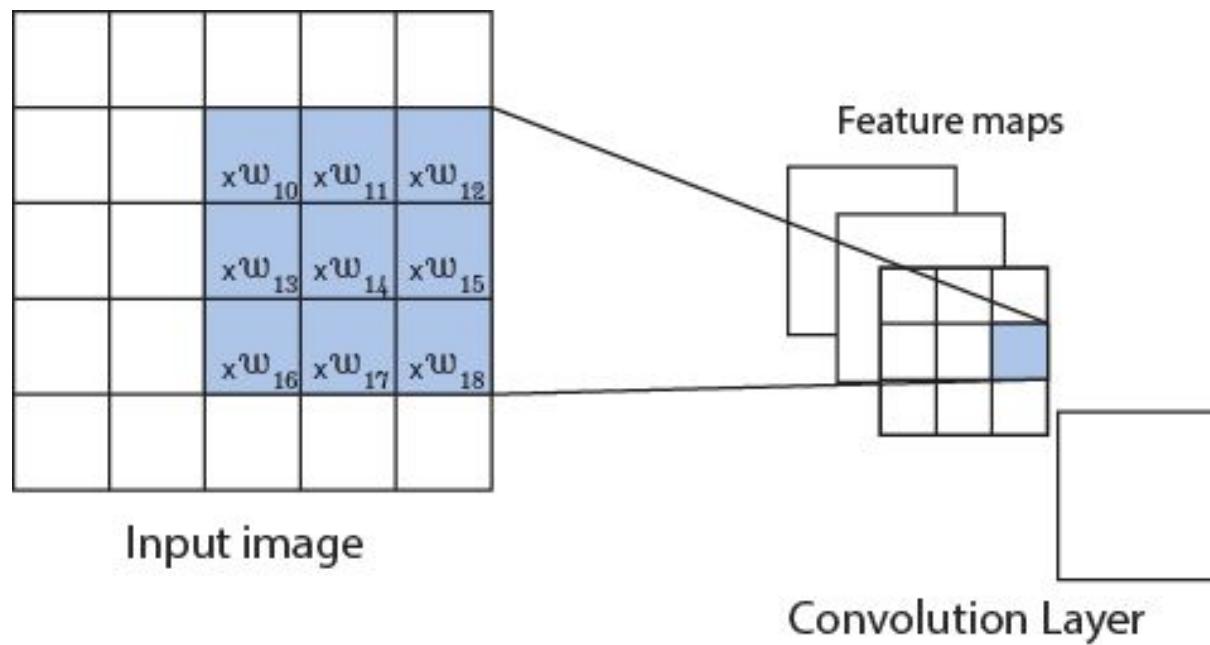
CNN - Example - 2nd Feature Map



CNN - Example - 2nd Feature Map



CNN - Example - 2nd Feature Map



CNN - Example - Consecutive Convolutions

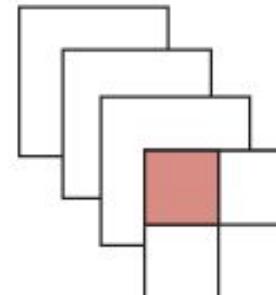
- Each filter in above layer performs convolution on all filters in previous layer, same for colour channels.

2x2 Kernel → 1 Stride

xw_{05}	xw_{06}	
xw_{07}	xw_{08}	

xw_{00}	xw_{01}	
xw_{03}	xw_{04}	

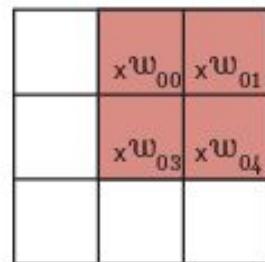
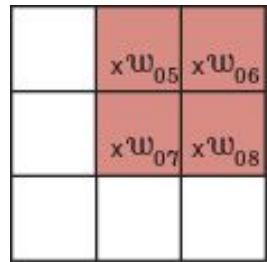
Convolution with
2 feature maps



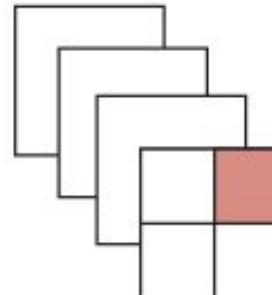
Convolution Layer

CNN - Example - Consecutive Convolutions

- Each filter in above layer performs convolution on all filters in previous layer, same for colour channels.



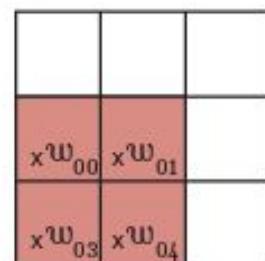
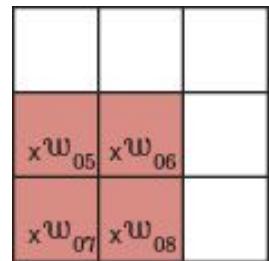
Convolution with
2 feature maps



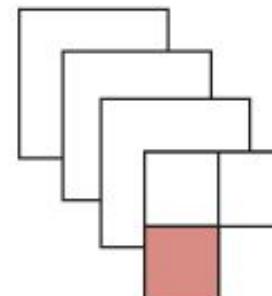
Convolution Layer

CNN - Example - Consecutive Convolutions

- Each filter in above layer performs convolution on all filters in previous layer, same for colour channels.



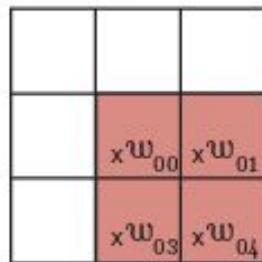
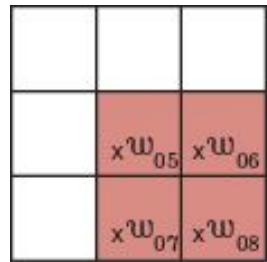
Convolution with
2 feature maps



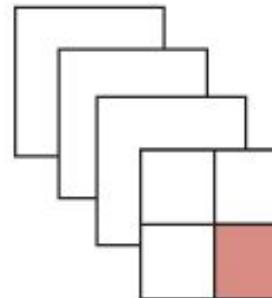
Convolution Layer

CNN - Example - Consecutive Convolutions

- Each filter in above layer performs convolution on all filters in previous layer, same for colour channels.



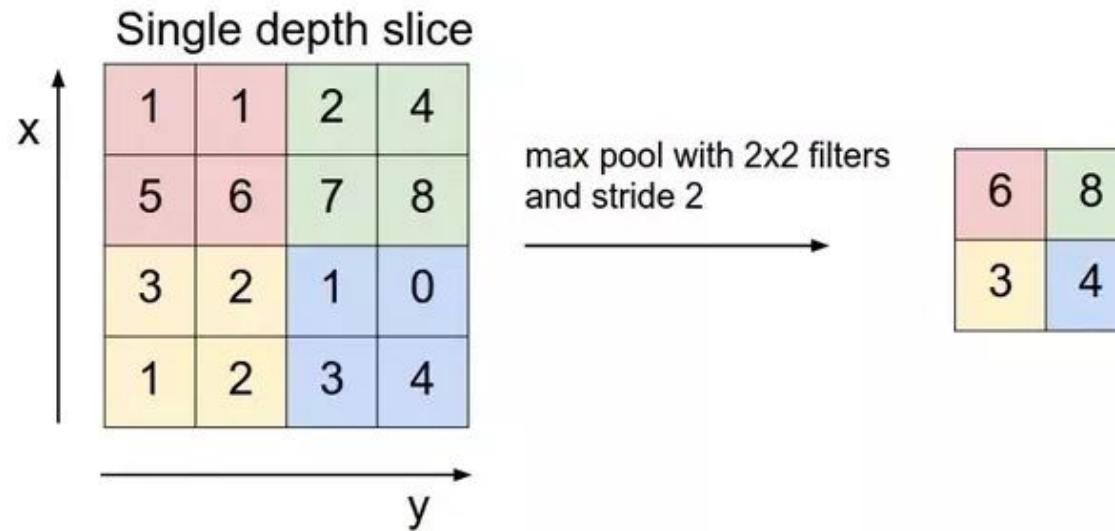
Convolution with
2 feature maps



Convolution Layer

Pooling

- Pooling performs subsampling and reduces network size
- Example of MAX pooling (selecting the maximum value)



[<http://cs231n.github.io/>]