PYTORCH

PyTorch: A flexible approach for computer vision models

Mo Patel @mopatel
Neejole Patel @datajolie

O'REILLY® Artificial Intelligence CONFERENCE
New York April 2018

Slack Channel for Tutorial:
https://pytorchcvtutorial.slack.com/
Invite Link: http://bit.ly/pytorchcvslack
Password: #AIConfNY2018

http://bit.ly/aiconfny18cv

Introductions & Networking

Agenda

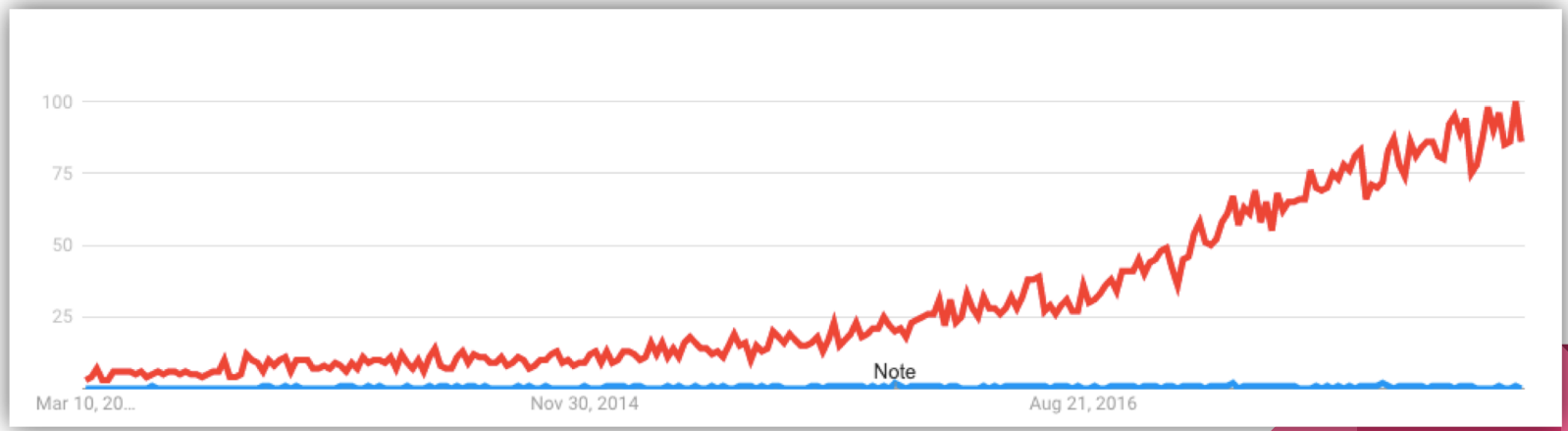
- PyTorch Overview
- Computer Vision Overview
- Getting started with PyTorch
- Break
- Computer Vision Models using PyTorch
  - Image Classification
  - Transfer Learning
  - Object Detection
  - Object Segmentation
- Computer Vision Datasets
- Deploying PyTorch Models
- PyTorch & Computer Vision Next

PyTorch Overview

PyTorch Origins

- Torch – Lua
- Chainer
- HIPS Autograd

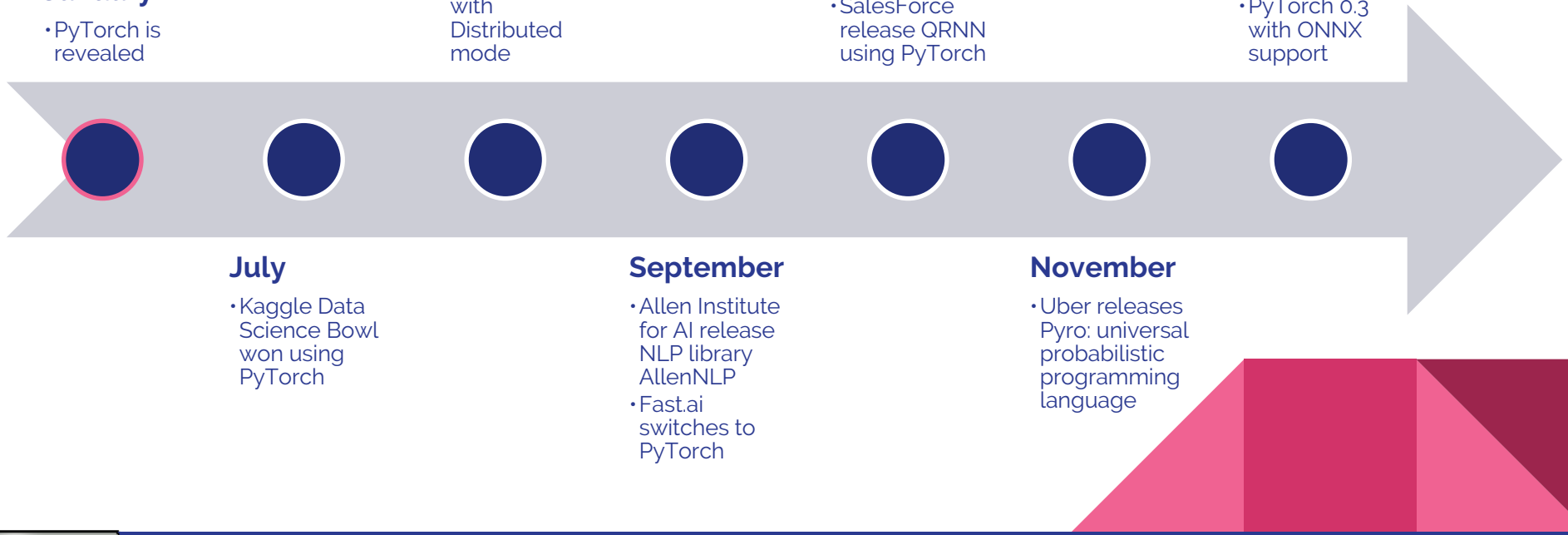
- Need for Dynamic execution library
- Popularity of Python in Machine Learning Community



Google Trends Last Five Years
Terms: Lua Machine Learning Python Machine Learning

http://bit.ly/aiconfny18cv

PyTorch – Year 2017 in Review



Source: http://pytorch.org/2018/01/19/a-year-in.html

PyTorch 2018

- April 2018: PyTorch 0.4
  - Tensors
    - Full support for advanced indexing
    - Fast Fourier Transforms
  - Neural Networks
    - Trade-off memory for compute
    - bottleneck - a tool to identify hotspots in your code
  - torch.distributions
    - 24 basic probability distributions
    - Added cdf, variance, entropy, perplexity etc.
  - Distributed Training
    - Launcher utility for ease of use
    - NCCL2 backend
  - C++ Extensions
  - **Windows Support**
  - ONNX Improvements
    - RNN support

http://pytorch.org/2018/04/22/0_4_0-migration-guide.html

PyTorch – Community Overview

By the numbers

- 112,480 lines of Python code on github that import torch
- 4,959 repositories on Github that mention PyTorch in their name or description
- More than half a million downloads of PyTorch binaries. 651,916 to be precise.
- **5,400 users** wrote **21,500 posts** discussing 5,200 topics on PyTorch forums discuss.pytorch.org
- 131 mentions of PyTorch on Reddit's /r/machinelearning since the day of release. In the same period, TensorFlow was mentioned 255 times.

Innovation driver

- In the recent ICLR2018 conference submissions, PyTorch was mentioned in **87 papers**, compared to TensorFlow at 228 papers, Keras at 42 papers, Theano and Matlab at 32 papers.
- Francois Chollet's monthly arxiv.org mentions for frameworks had PyTorch at 72 mentions, with TensorFlow at 273 mentions, Keras at 100 mentions, Caffe at 94 mentions and Theano at 53 mentions.

Source: http://pytorch.org/2018/01/19/a-year-in.h

Use Case Diversity: Research & Applications

PyTorch Comparison

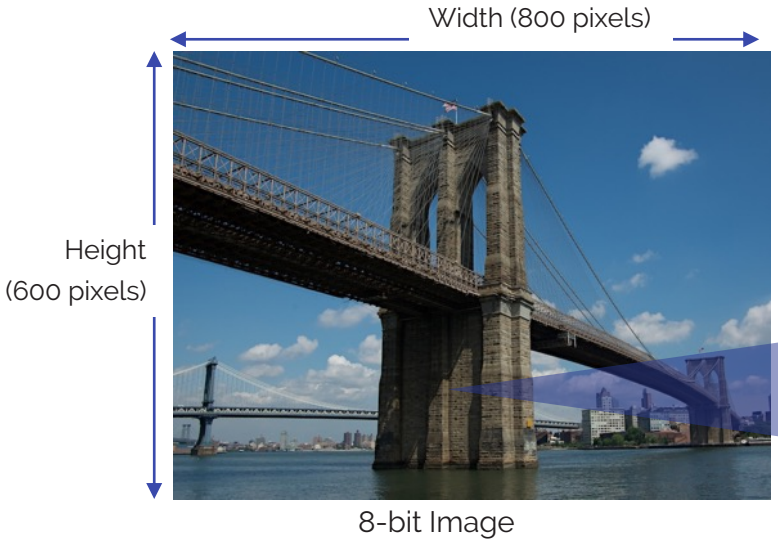
- PyTorch dynamic computation allows flexibility of input and Python style debugging
- PyTorch best suited for research and prototyping, deployment via ONNX to other frameworks such as Caffe2, Apple CoreML, MXNet, Tensorflow
- Seasoned users of Python data stack (NumPy) can easily transition to PyTorch

Source: https://awni.github.io/pytorch-tensorflow/

Computer Vision Overview

Computer Vision 101

- Image Properties
  - Pixels
  - Dimension
  - Channels
  - Color Depth



8-bit Image

Color Depth	Channel Range
1-Bit	0-1
8-Bit	0-255
16-Bit	0-65,535
24-Bit	0-16,777,215

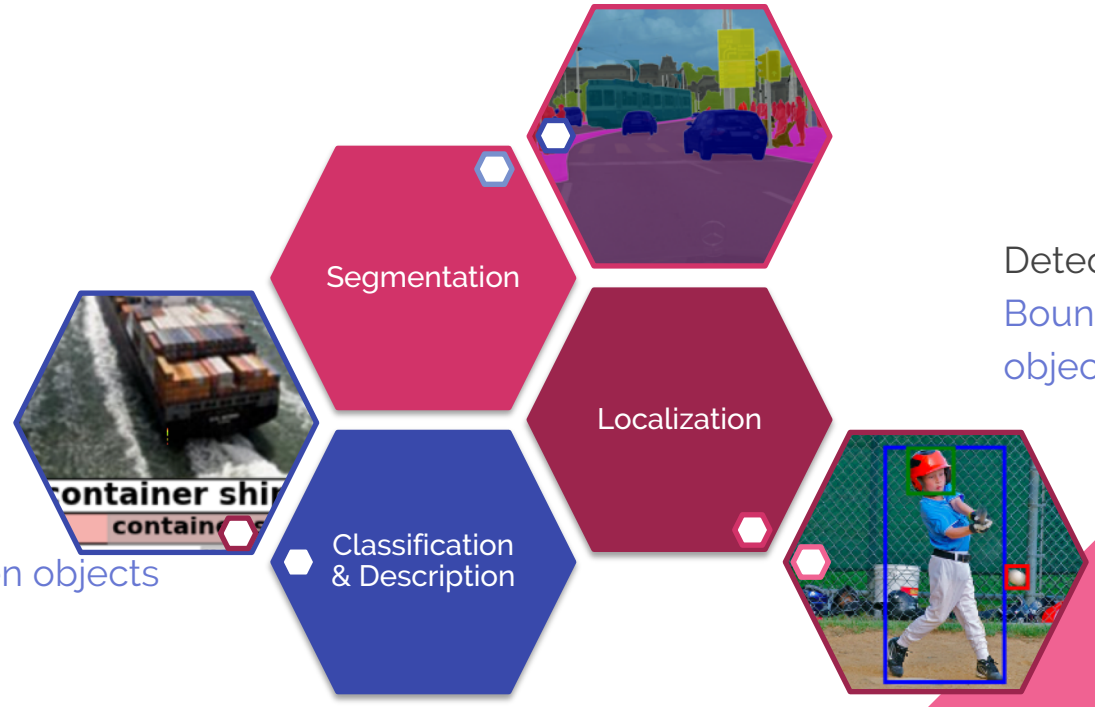
One Pixel

Channel	Red	Green	Blue
Value	0-255	0-255	0-255

http://bit.ly/aiconfny18cv

Computer Vision 101

Segmentation: Pixel-wise segments of objects in any image



Classification: Label Item(s) in Image

Description: Relationship between objects

Detection/Localization: Bounding Boxes around objects

Goal: Build models that can perform visual tasks

Classification & Description

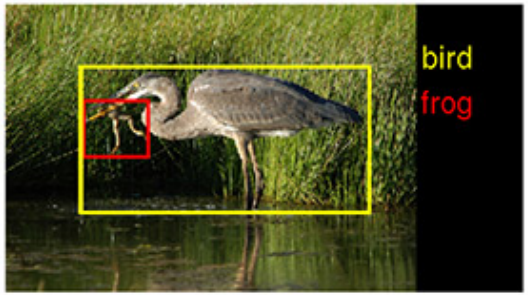
What is the man doing?

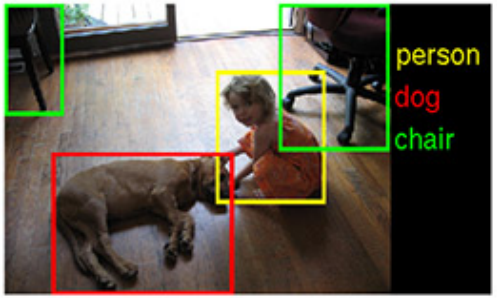
Submit

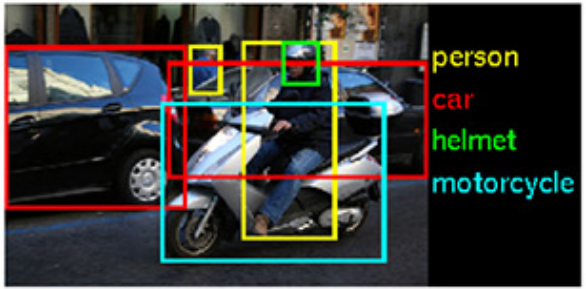
Predicted top-5 answers with confidence:

playing	99.126%
tennis	
tennis	0.553%
skateboarding	0.139%
serving	0.075%
jumping	0.033%

Localization







Source: Kaggle ImageNet Object Detection Challenge

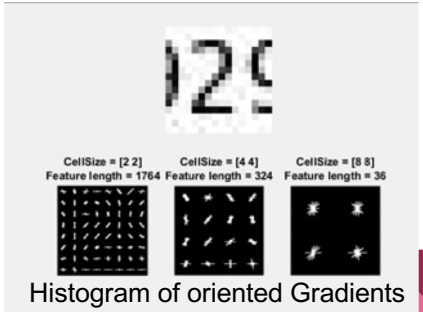
Segmentation

Source: The Cityscapes Dataset - Semantic, instance-wise, dense pixel annotations of 30 classes

Computer Vision prior to Deep Learning

- Objectives
  - Extract generalized features (patterns) from images
  - Determine generalized descriptors for feature
  - Use generalized features to perform visual tasks based on descriptors

- What are features?
  - Edges
  - Shapes
  - Change in Colors/Intensity



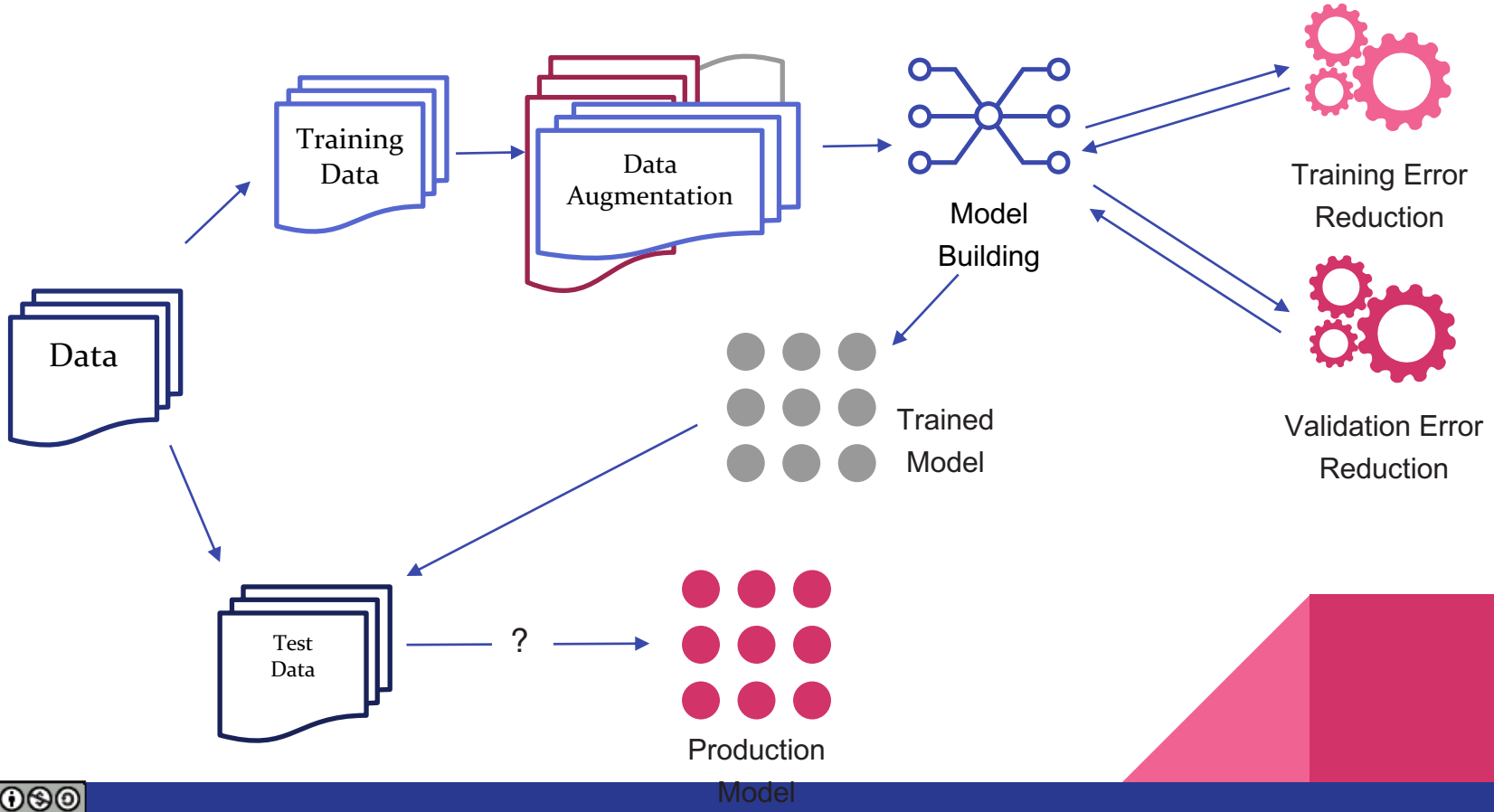
https://www.mathworks.com/products/computer-vision/features.html

Traditional Machine Learning → Deep Learning

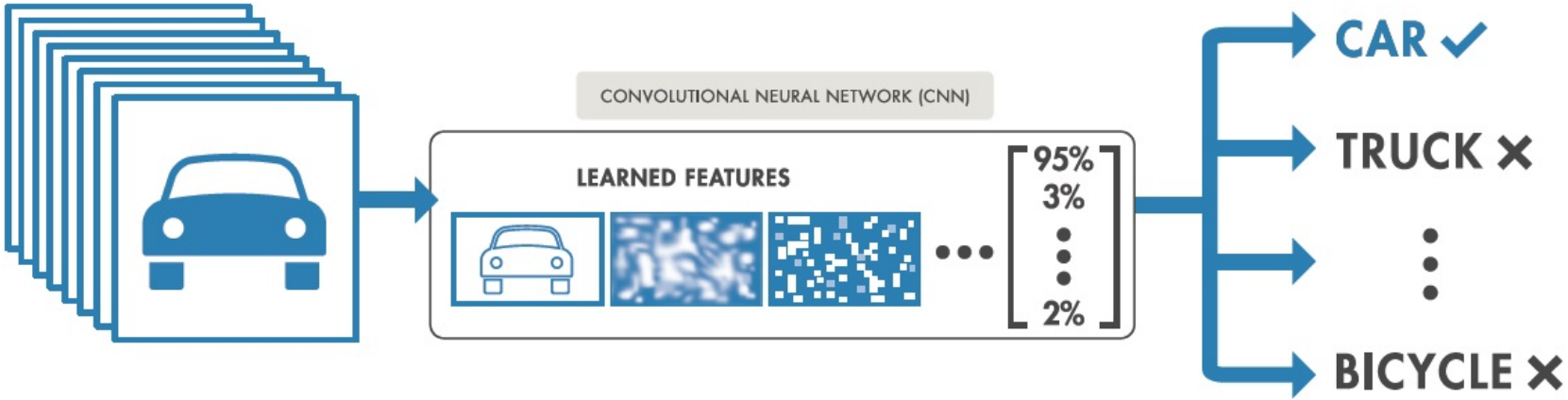
- Featuring engineering challenges
  - Human scaling: Time consuming to create rules for each object
  - Computational scaling: Translating human curated features to algorithms that machines can perform at scale is tedious
- Feature Learning
  - Design computational models to learn features
- Feature Learning Challenges
  - Need large datasets with annotations
  - Model architectures become complex both to design and run

Convolutional Neural Networks

Deep (Supervised Machine) Learning Basics



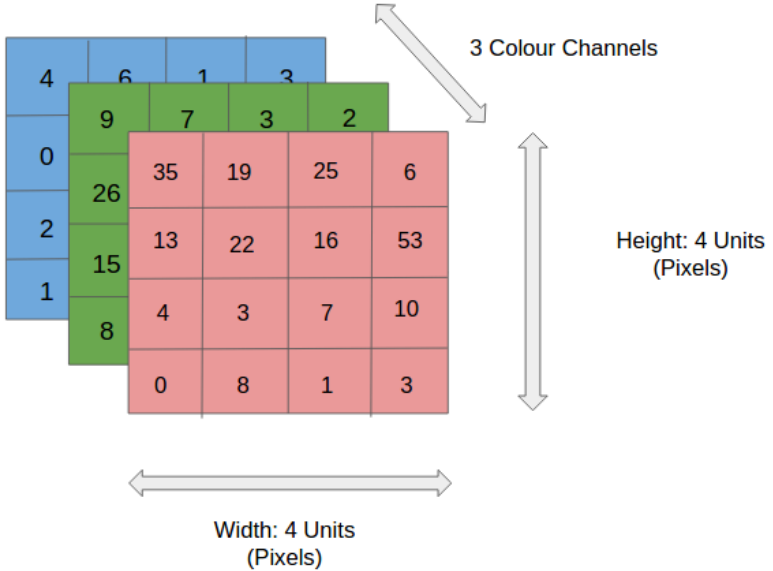
Convolutional Neural Networks



Source: Mathworks

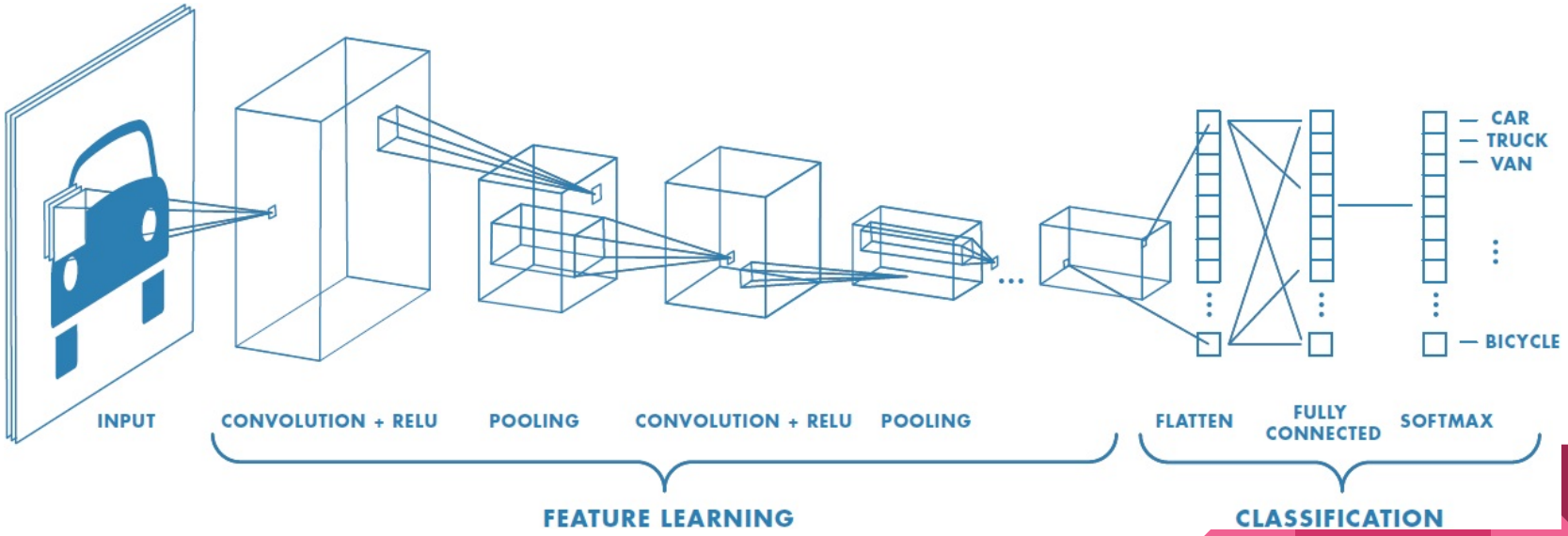
Convolutional Neural Network Properties

- Learning Low Level Features from High Dimensional Data
- Must Have Data Properties:
  - Shift Invariant
  - Correlated
- Simply put: Changing column order should not change meaning of Data
- Naturally Suitable Data Types:
  - Images, Audio, Video
  - Time Series



http://bit.ly/aiconfny18cv

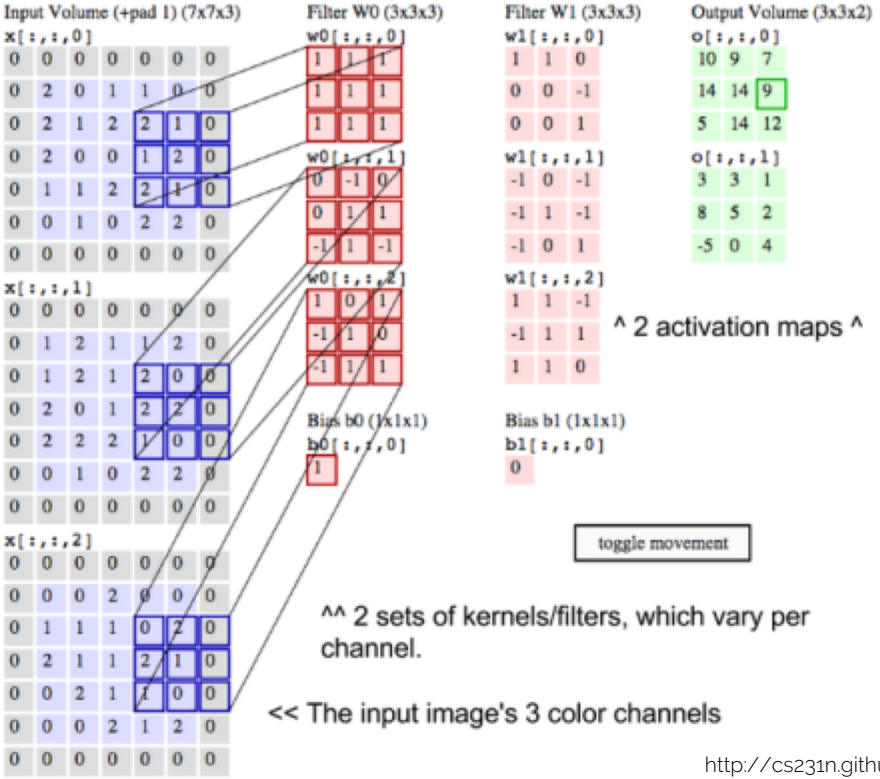
Convolutional Neural Networks

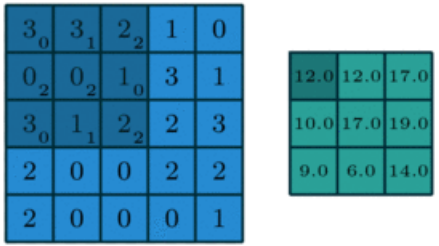


Source: Mathworks

http://bit.ly/aiconfny18cv

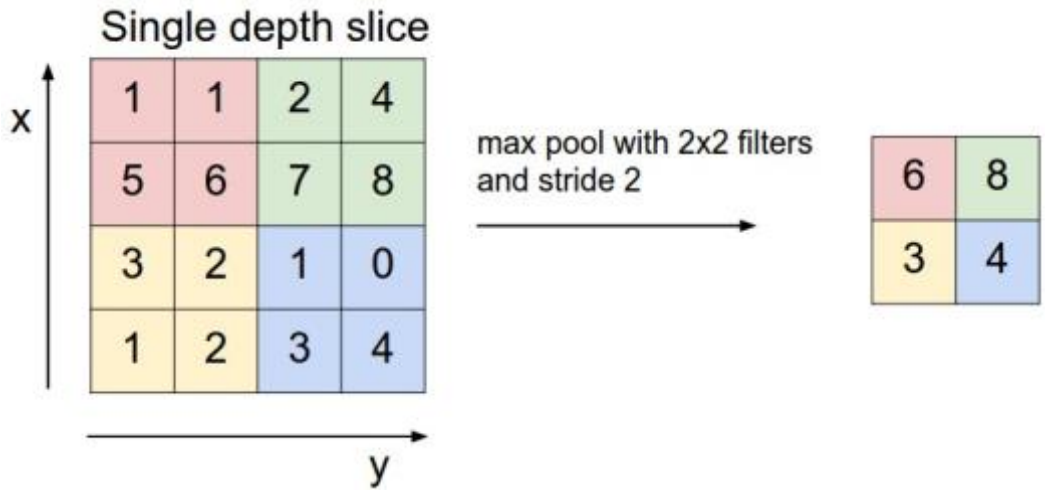
Convolution





http://cs231n.github.io/convolutional-networks/
https://m2dsupsdlclass.github.io/lectures-labs/slides/03_conv_nets/index.html
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

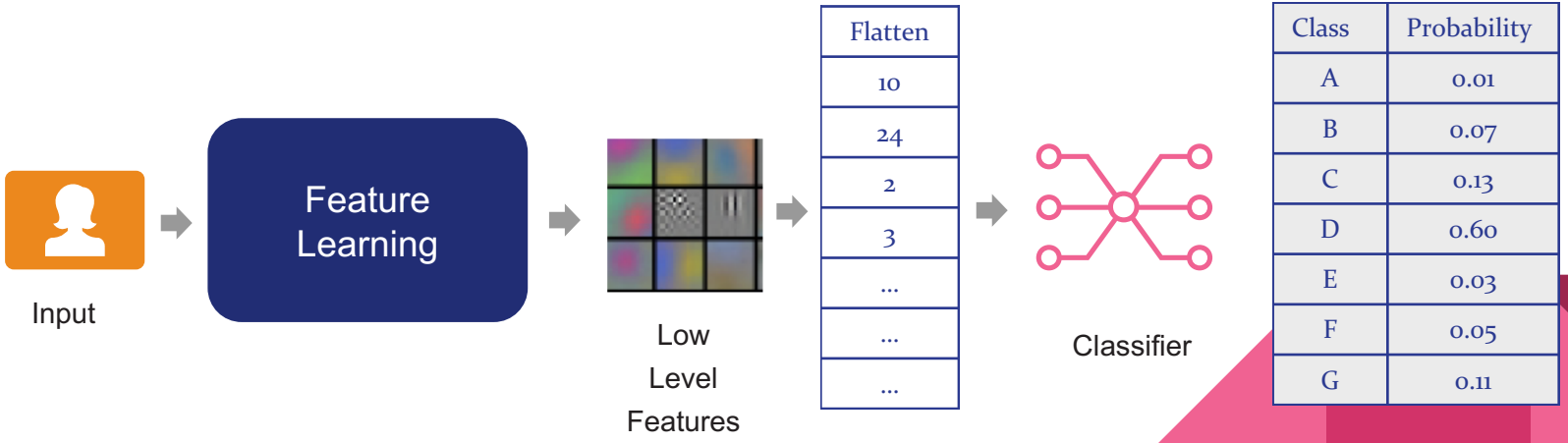
Pooling



Source: Stanford University CS 231 Course:
http://cs231n.github.io/convolutional-networks/#pool

Fully Connected

- Traditional Machine Learning: Feature Engineering
- Deep Learning: Feature Learning
- Learned Features are fed to traditional classifier such as Multivariate Logistic Regression (Softmax)



Loss Reduction

Class	Probability	Truth	Loss/ Error
A	0.01	0	0
B	0.07	0	0
C	0.13	0	0
D	0.60	1	0.22
E	0.03	0	0
F	0.05	0	0
G	0.11	0	0

Cross Entropy Loss

One-Hot Encoded Vector of Class Label

Back Propagation

- Account for loss across the weights and biases in the network
- Use cost function provided by Back Propagation to optimize the weights and biases to reduce loss
- Variety of Stochastic Gradient Descent techniques used for optimization

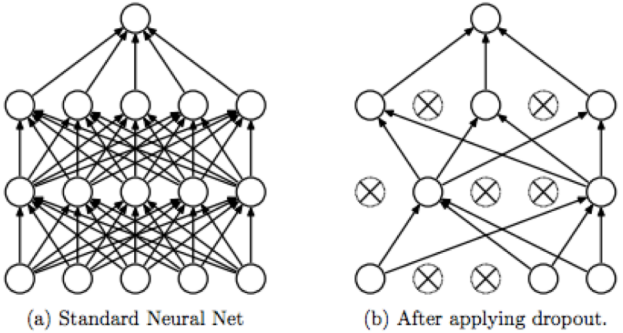
Overview of Gradient Descent Techniques:
http://ruder.io/optimizing-gradient-descent/

https://rdipietro.github.io/friendly-intro-to-cross-entropy-loss/

Generalization

Dropout

Randomly remove learned nodes from network



(a) Standard Neural Net (b) After applying dropout.

Dropout: A Simple Way to Prevent Neural Networks from Overfitting
http://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Batch Normalization

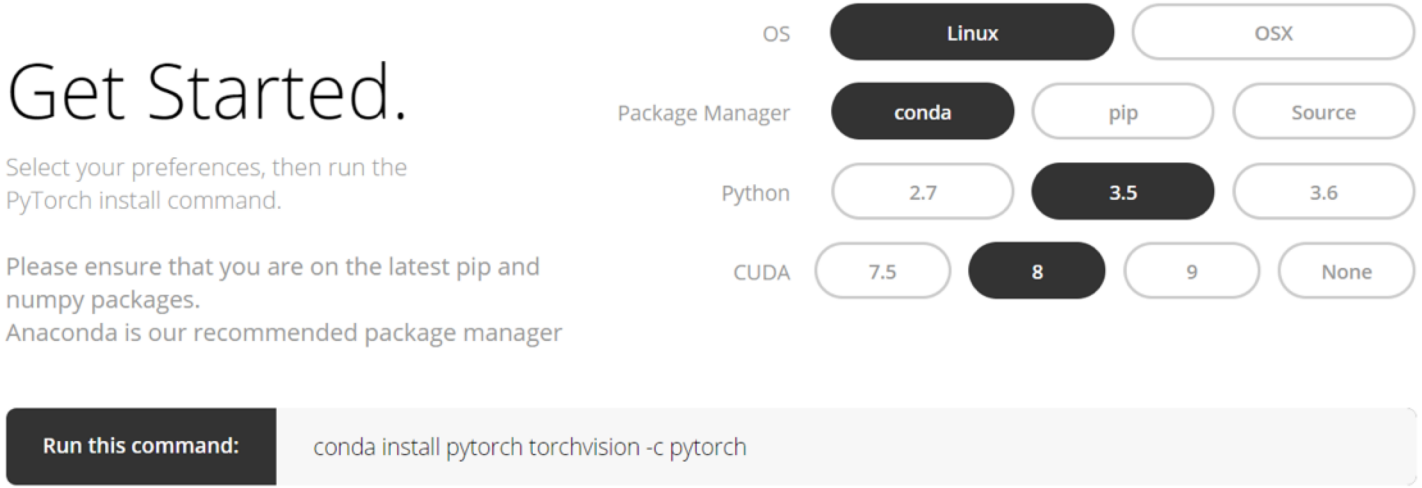
- During training input data shifts as weights and parameter adjust values
- To prevent internal covariate shift, normalize each batch by mean and variance

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift
https://arxiv.org/abs/1502.03167

Getting started with PyTorch

Installation

http://pytorch.org/



PyTorch Basics Jupyter Notebook

- Using Jupyter Notebook, please open *pytorch_basics.ipynb* from the tutorial repository on Github.

Automatic Differentiation

Automatic Differentiation

Chain rule for today's computational needs:

1. Technique to determine derivative of function within a function
2. Complex functions can be written as compositions of simple functions

- PyTorch `autograd` package: Provides automatic differentiation on all tensor operations

torch.Tensor

- Central class for autograd
  - Wraps a tensor and records the history use `requires_grad=True`
  - Supports all tensor operations
- Use .backward
  - Gradients computed automatically

PyTorch Autograd Notebook

- Using Jupyter Notebook, please open *autograd_tutorial.ipynb* from the tutorial repository on Github.

PyTorch
Neural Net & Optimization
Package Overview

PyTorch's nn module

- Containers
- Convolution Layers
- Pooling Layers
- Padding Layers
- Non-linear Activations (weighted sum + nonlinearity)
- Non-linear Activations (Other)
- Normalization layers
- Recurrent layers
- Linear layers
- Dropout layers
- Sparse layers
- Distance functions
- Loss functions
- Vision layers
- DataParallel layers (multi-GPU, distributed)
- Utilities

PyTorch's optim algorithms

Algorithms		Algorithms	
adadelta	Adaptive Delta	sgd	Stochastic Gradient Descent
adagrad	Adaptive Subgradients	asgd	Averaged Stochastic Gradient Descent
adam	Stochastic Optimization	lbfgs	Limited BFGS
adamax	Adam based on Infinity Norm	rprop	Resilient Backpropagation
sparse_adam	Adam suitable for sparse tensors	rmsprop	Root Mean Squared Propagation

TorchVision

PyTorch Torchvision

- Datasets
  - MNIST
  - Fashion-MNIST
  - EMNIST
  - COCO
  - LSUN
  - Imagenet-12
  - CIFAR
  - STL10
  - SVHN
  - PhotoTour
- Utilities
  - make_grid
  - save_image
- Models
  - Alexnet
  - VGG
  - ResNet
  - SqueezeNet
  - DenseNet
  - Inception v3
- Transforms

Break