



# INTRODUCTION TO DEEP LEARNING

MUSTAFA ALDEMIR, INTEL TURKEY



ARTIFICIAL  
INTELLIGENCE

# WHAT IS DEEP LEARNING GOOD FOR

# DEEP LEARNING: EXAMPLES



## Images

Computer vision, Image classification,  
Traffic sign detection, Pedestrian  
detection, localization...



## Sound

Speech recognition, Natural Language  
Processing, Translation, Content  
captioning, speaker identification

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

## Text

Natural Language Processing, text  
classification; web search, spam,  
email filtering

# CLASSIFICATION

-> Label the image

Person

Motorcyclist

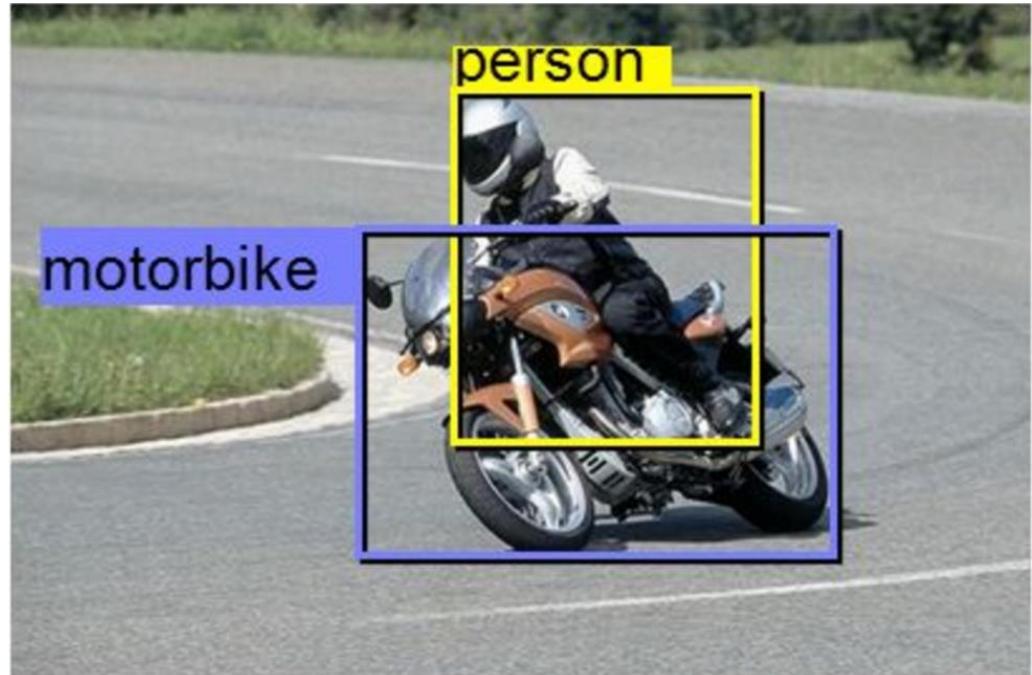
Bike



<https://people.eecs.berkeley.edu/~jhoffman/talks/llda-baylearn2014.pdf>

# DETECTION

-> Detect and label



<https://people.eecs.berkeley.edu/~jhoffman/talks/llda-baylearn2014.pdf>

# SEMANTIC SEGMENTATION

-> Label every pixel



<https://people.eecs.berkeley.edu/~jhoffman/talks/llda-baylearn2014.pdf>

# NATURAL LANGUAGE OBJECT RETRIEVAL

a scene with three people

query='man far right'



query='left guy'

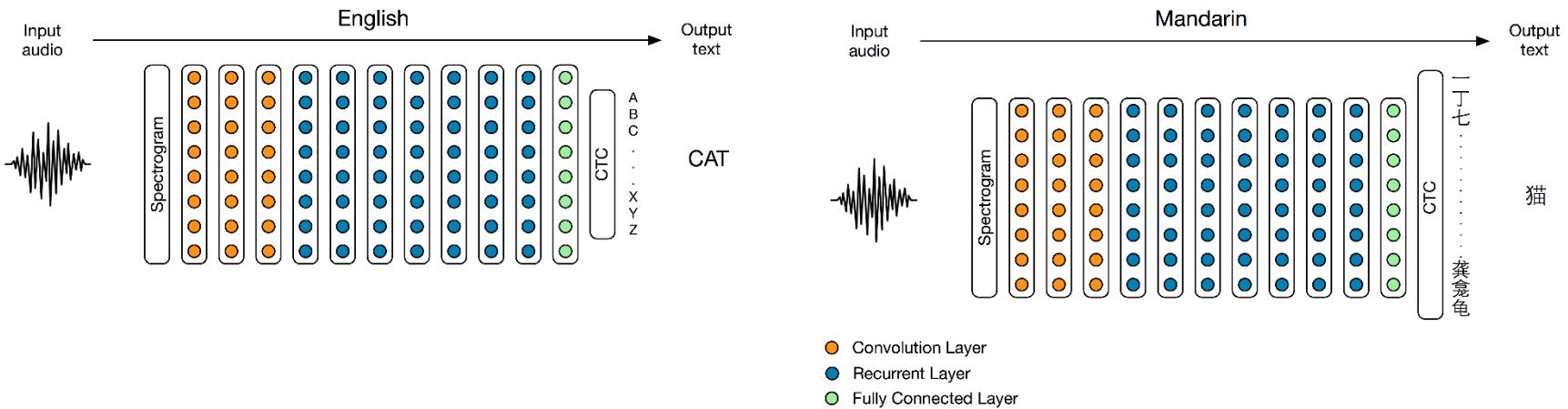


query='cyclist'



<http://arxiv.org/pdf/1511.04164v3.pdf>

# SPEECH RECOGNITION

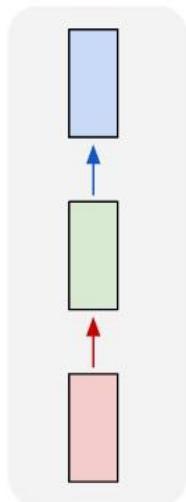


# IMAGE / VIDEO CAPTIONING

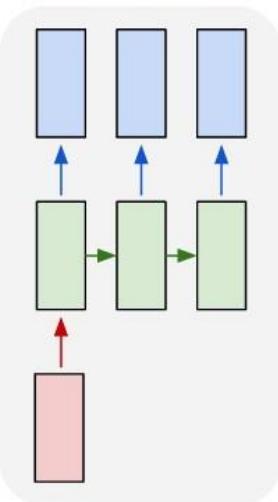
| Describes without errors   | Describes with minor errors   | Somewhat related to the image   |
|--|---|---|
|  A person riding a motorcycle on a dirt road.       |  Two dogs play in the grass.                    |  A skateboarder does a trick on a ramp.          |
|  A group of young people playing a game of frisbee. |  Two hockey players are fighting over the puck. |  A little girl in a pink hat is blowing bubbles. |

# RECURRENT NEURAL NETWORKS

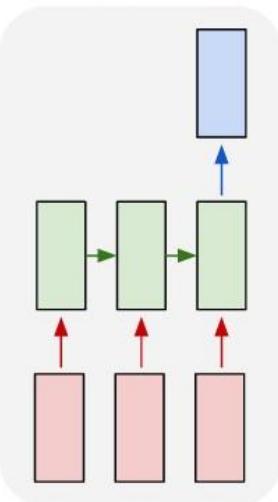
one to one



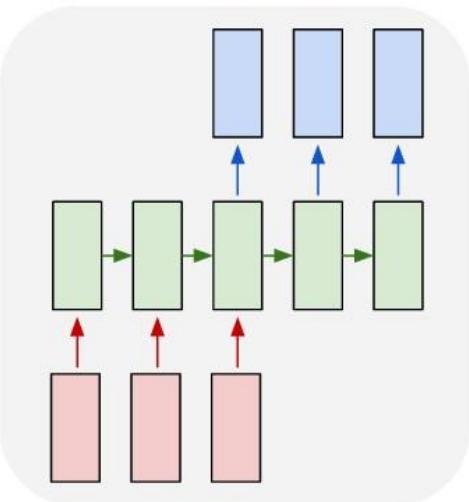
one to many



many to one



many to many



many to many

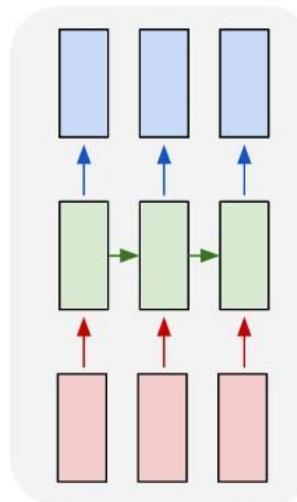


Image  
Classification

Image  
Captioning

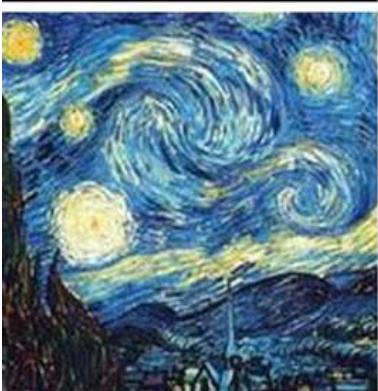
Sentiment  
Analysis

Google Translate

Video Frame  
Classification

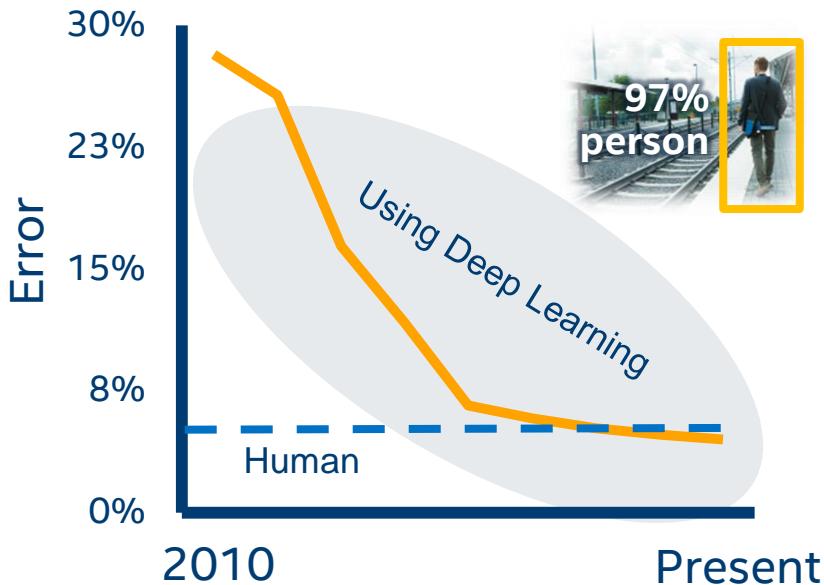
Red: input; Blue: output, Green: RNN state

# GENERATIVE ADVERSARIAL NETWORKS

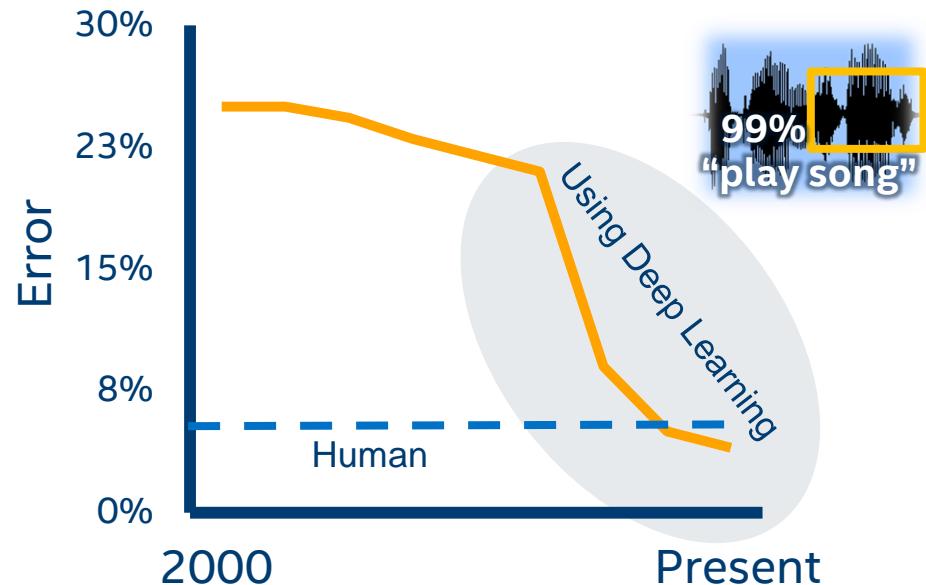


# DEEP LEARNING BREAKTHROUGHS

## IMAGE RECOGNITION



## SPEECH RECOGNITION

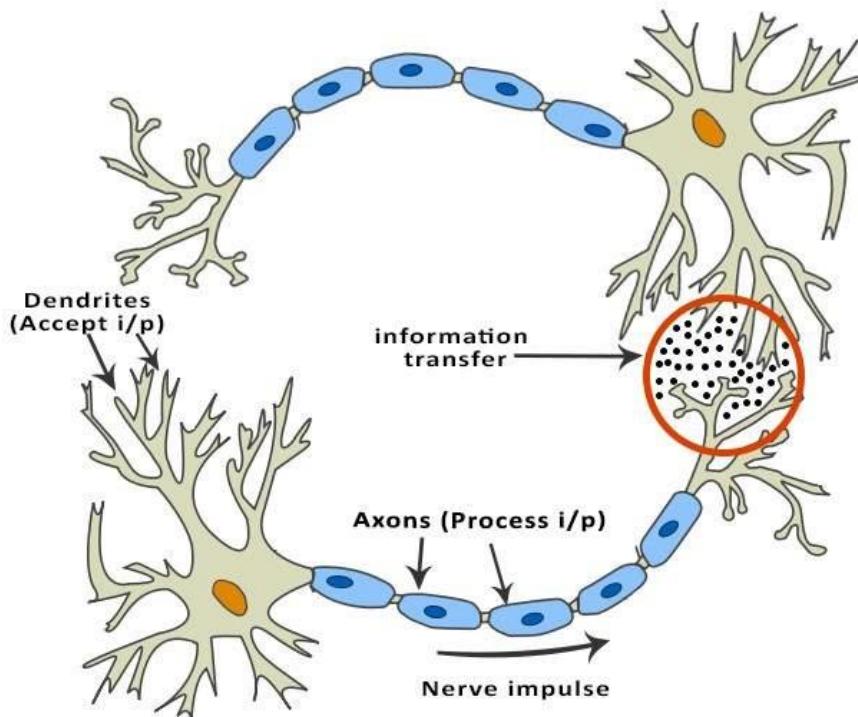


These and more are enabling new & improved applications



# NEURAL NETWORKS

# INSPIRED BY HUMAN BRAIN



# CASE STUDY: RECOMMENDATION SYSTEMS

A screenshot of the Netflix homepage. At the top, there's a navigation bar with the Netflix logo, a "Browse" dropdown, a "DVD" link, a search bar with a magnifying glass icon, and a user profile for "Mr. Wadhwa". Below the navigation is a banner with five TV show thumbnails: "SHERLOCK", "THE X-FILES", "THE RETURNED", "NURSE JACKIE", and "UNBREAKABLE KIMMY SCHMIDT". Underneath this banner, there's a section titled "TV Shows" with five more thumbnails: "LOVE", "BETTER CALL SAUL", "COOKED", "BATES MOTEL", and "HIGHWAY THRU HELL".

NETFLIX

Browse ▾ DVD

Continue Watching for Mr. Wadhwa

Search

Mr. Wadhwa ▾

SHERLOCK

THE X-FILES

THE RETURNED

NEW EPISODES

NURSE JACKIE

UNBREAKABLE KIMMY SCHMIDT

TV Shows

LOVE

BETTER CALL SAUL

COOKED

BATES MOTEL

HIGHWAY THRU HELL

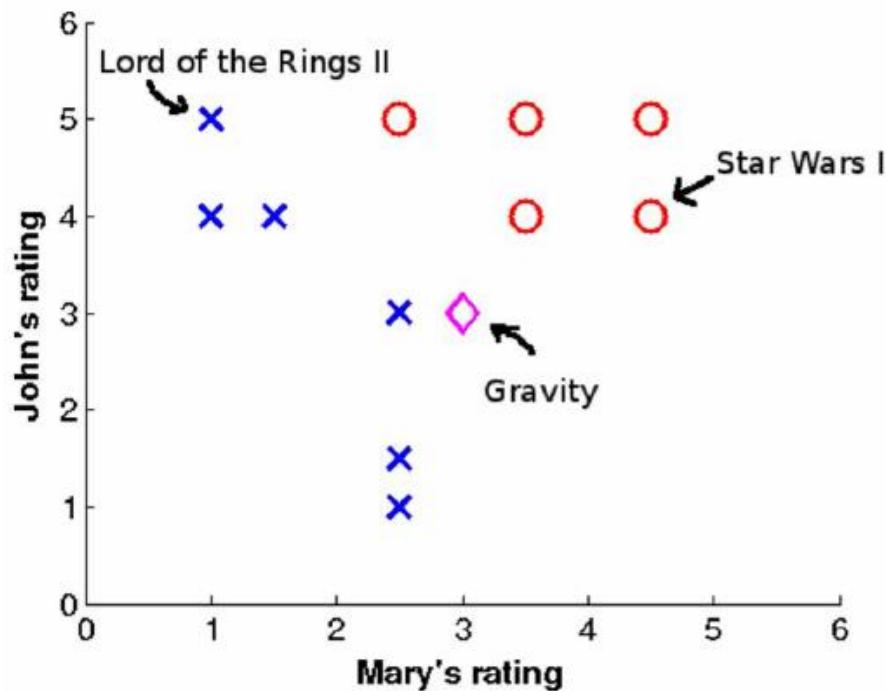


Will Nancy like Gravity?

Let's ask close friends Mary and John, who already watched it and rated between 1-5.



| Movie               | Mary's Rating | John's Rating | Does Nancy like? |
|---------------------|---------------|---------------|------------------|
| Lord of the Rings 2 | 1             | 5             | No               |
| ...                 | ...           | ...           | ...              |
| Star Wars 1         | 4.5           | 4             | Yes              |
| Gravity             | 3             | 3             | ?                |



A decision function can be as simple as weighted linear combination of friends:

$$h_{\theta,b} = \theta_1 x_1 + \theta_2 x_2 + b$$

$$h_{\theta,b} = \theta^T x + b$$

- Labels: “I like it” -> 1 “I don’t like it” -> 0
- Inputs: Mary’s rating, John’s rating

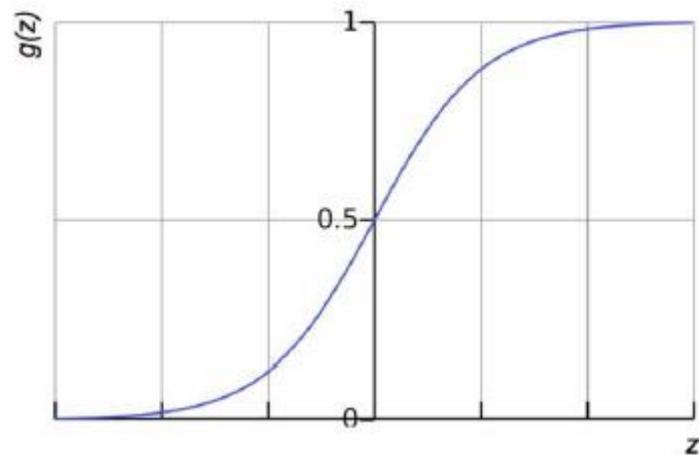
# ACTIVATION FUNCTION

This function has a problem. Its values are unbounded.  
We want its output to be in the range of 0 and 1.

$$h_{\theta,b} = g(\theta^T x + b),$$

where  $g(z)$  is sigmoid function.

$$g(z) = \frac{1}{1 + \exp(-z)}$$

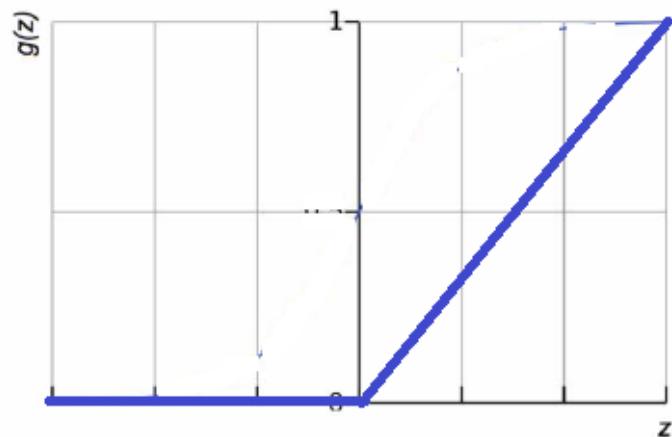


# ACTIVATION FUNCTION

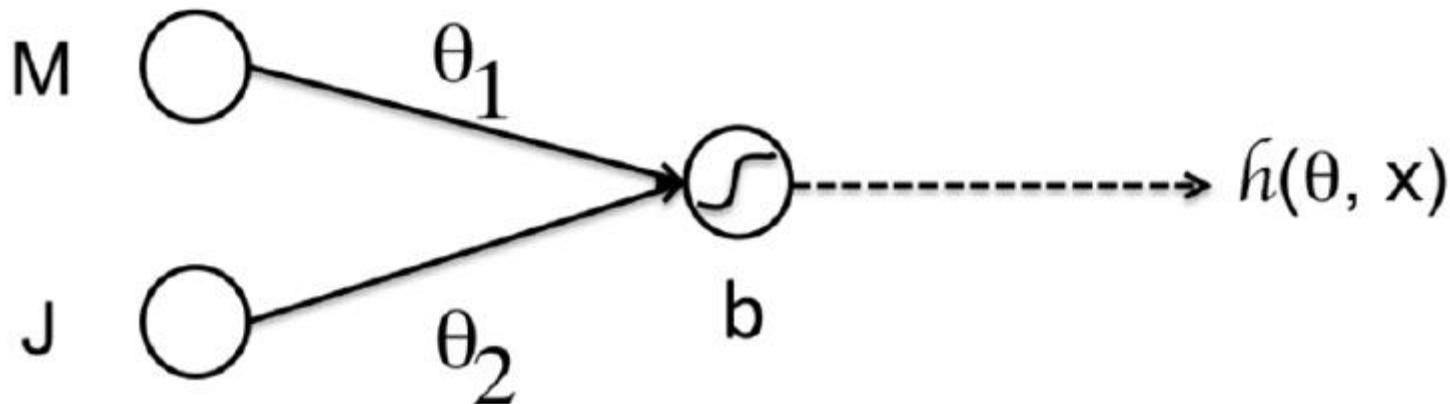
This function has a problem. Its values are unbounded.  
We want its output to be in the range of 0 and 1.

## ReLU (Rectified Linear Unit)

$$f(x) = \max(0, x)$$



## ANOTHER WAY OF REPRESENTING THE MODEL



# LEARN FROM DATA

We will use the past data to learn  $\theta, b$  to approximate  $y$ . In particular, we want to obtain  $\theta, b$  such that:

$h_{\theta,b}(x^{(1)}) \approx y^{(1)}$  where  $x^{(1)}$  is my friend's ratings for 1<sup>st</sup> movie.

$h_{\theta,b}(x^{(2)}) \approx y^{(2)}$  where  $x^{(2)}$  is my friend's ratings for 2<sup>nd</sup> movie.

...

$h_{\theta,b}(x^{(m)}) \approx y^{(m)}$  where  $x^{(m)}$  is my friend's ratings for m<sup>th</sup> movie.

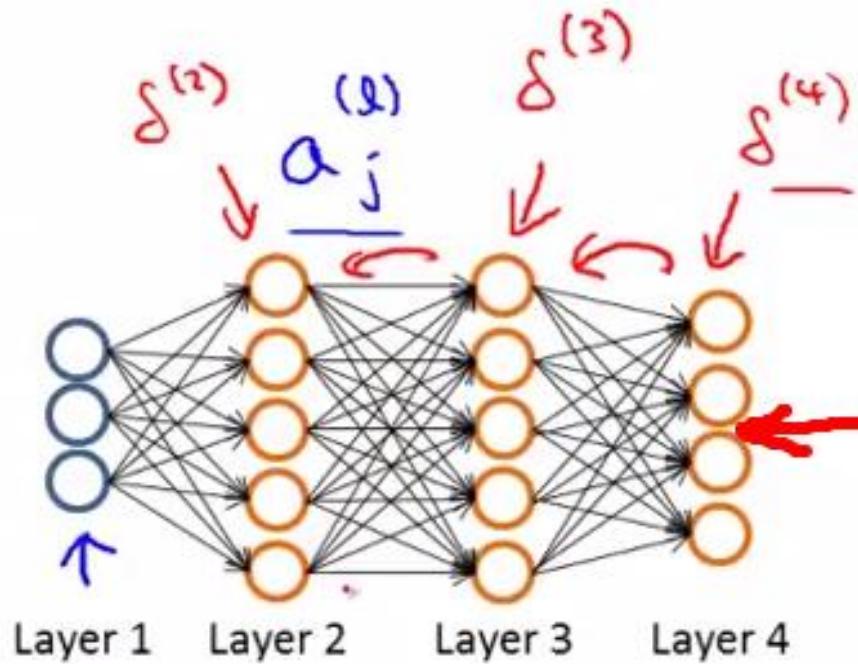
# COST FUNCTION

To find values of  $\theta$  and  $b$  we can minimize the following *cost function*:

$$J(\theta, b) = (h_{\theta,b}(x^{(1)}) - y^{(1)})^2 + (h_{\theta,b}(x^{(2)}) - y^{(2)})^2 + \dots + (h_{\theta,b}(x^{(m)}) - y^{(m)})^2$$

$$J(\theta, b) = \sum_{i=1}^m (h_{\theta,b}(x^{(i)}) - y^{(i)})^2$$

# BACKPROPOGATION



# STOCHASTIC GRADIENT DESCENT

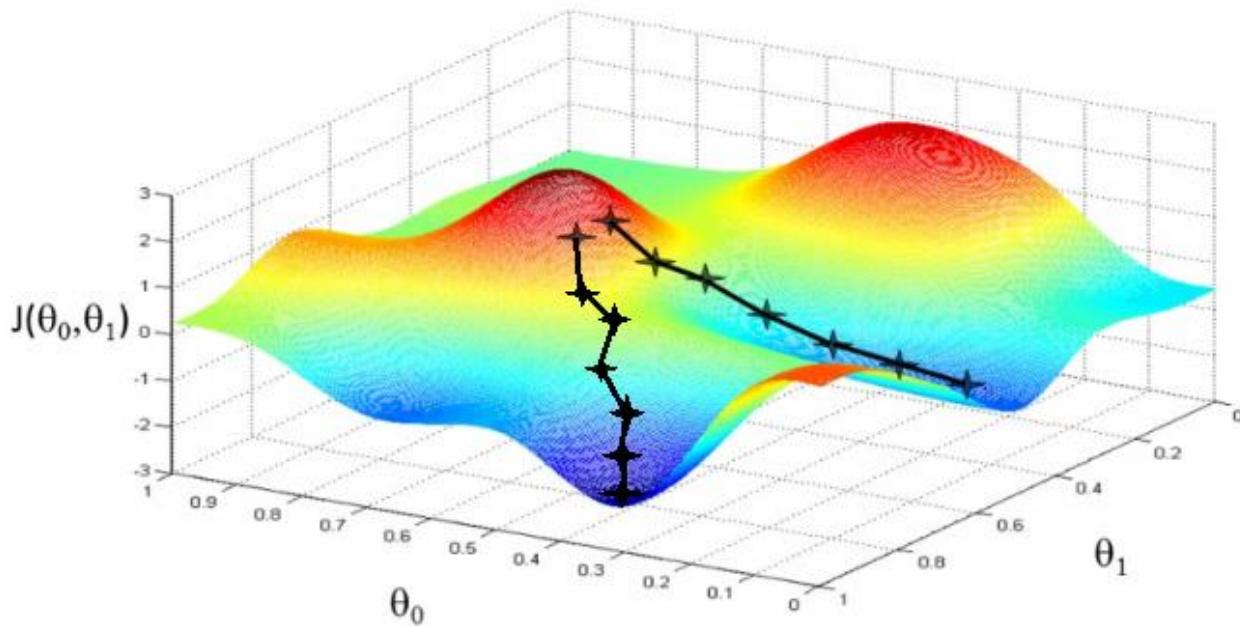
Use Stochastic Gradient Descent (SGD):

$$\theta_1 = \theta_1 - \alpha \Delta \theta_1$$

$$\theta_2 = \theta_2 - \alpha \Delta \theta_2$$

$$b = b - \alpha \Delta b$$

# STOCHASTIC GRADIENT DESCENT



# STEPS

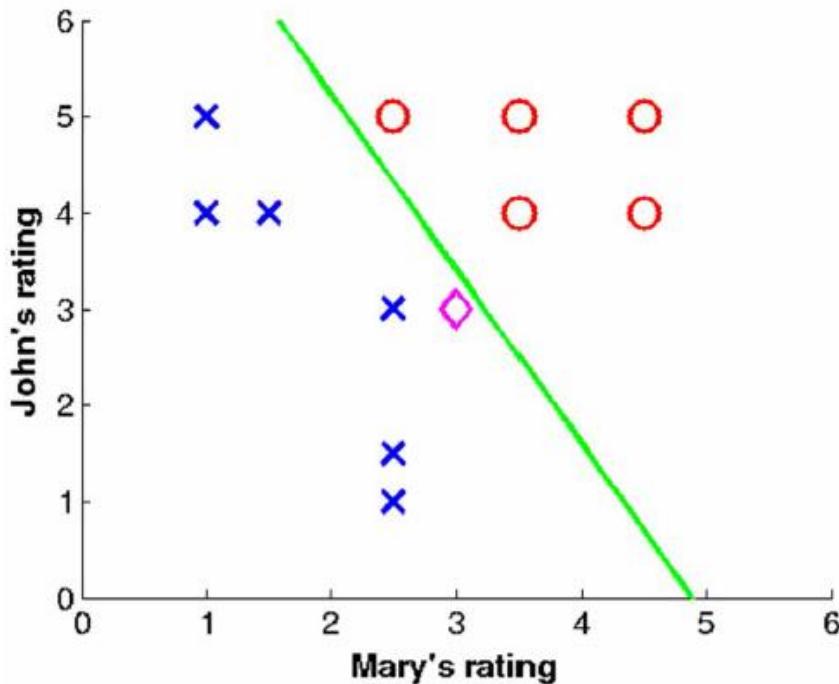
1. Initialize the parameters  $\theta$  and  $b$  at random
2. Pick a random example  $\{x^{(i)}, y^{(i)}\}$
3. Compute the partial derivatives of  $\theta_1, \theta_2, b$
4. Update parameters using:

$$\theta_1 = \theta_1 - \alpha \Delta \theta_1$$

$$\theta_2 = \theta_2 - \alpha \Delta \theta_2$$

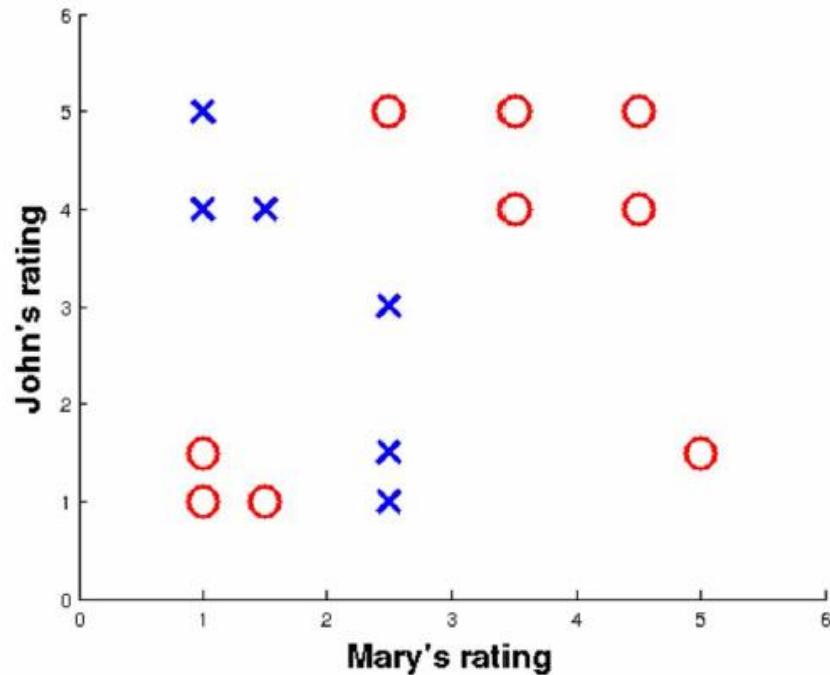
$$b = b - \alpha \Delta b$$

Stop it when parameters don't change much, or after a certain number of iterations.



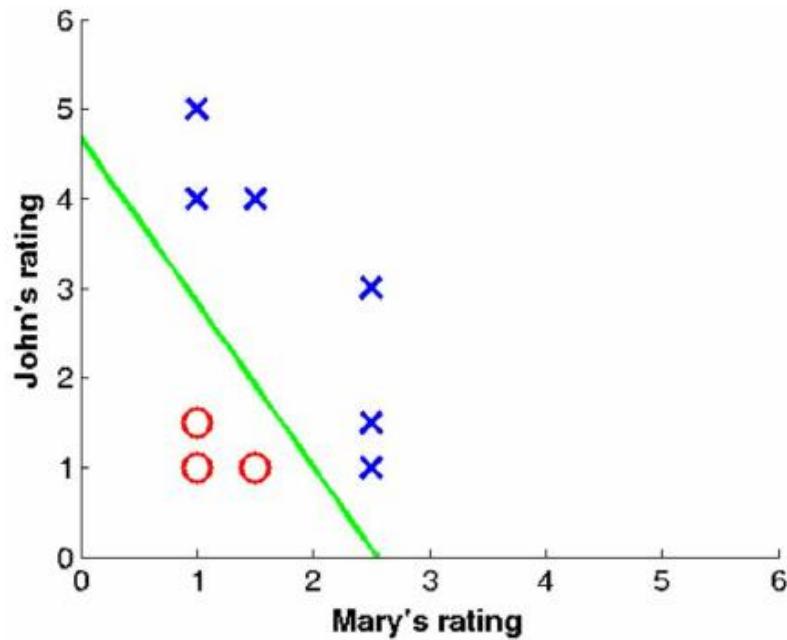
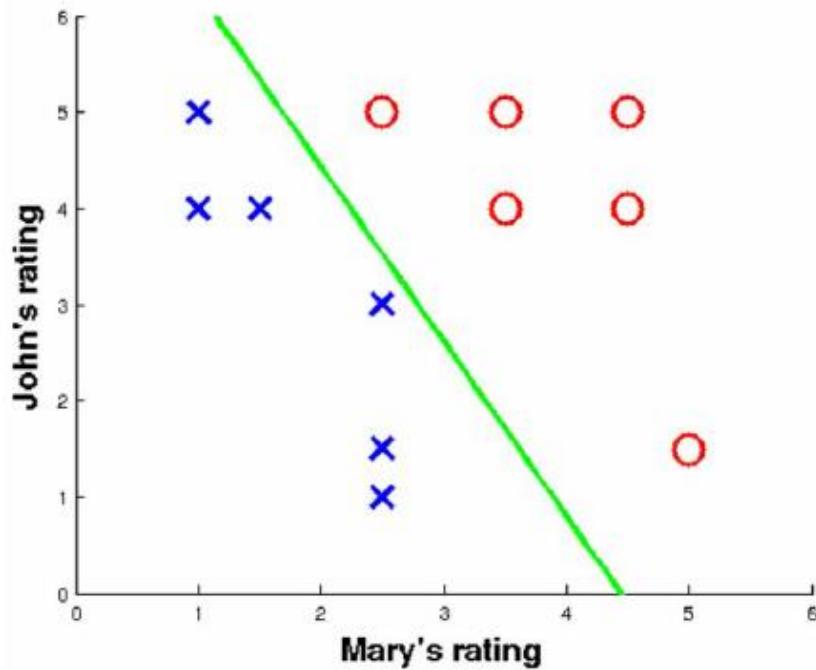
Gravity movie is slightly on the “don’t watch” side.

With this data set, it seems like “not watching it” makes more sense.



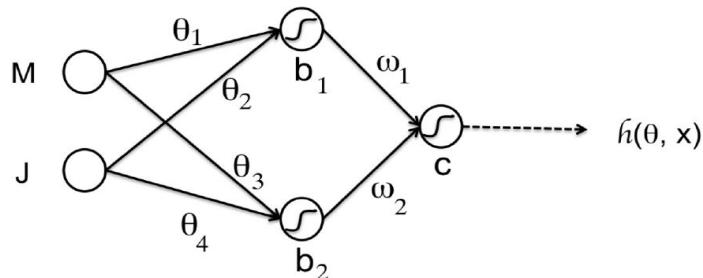
Nancy likes some of the movies both Mary and John rated poorly.

How can I have a linear decision boundary separate these?

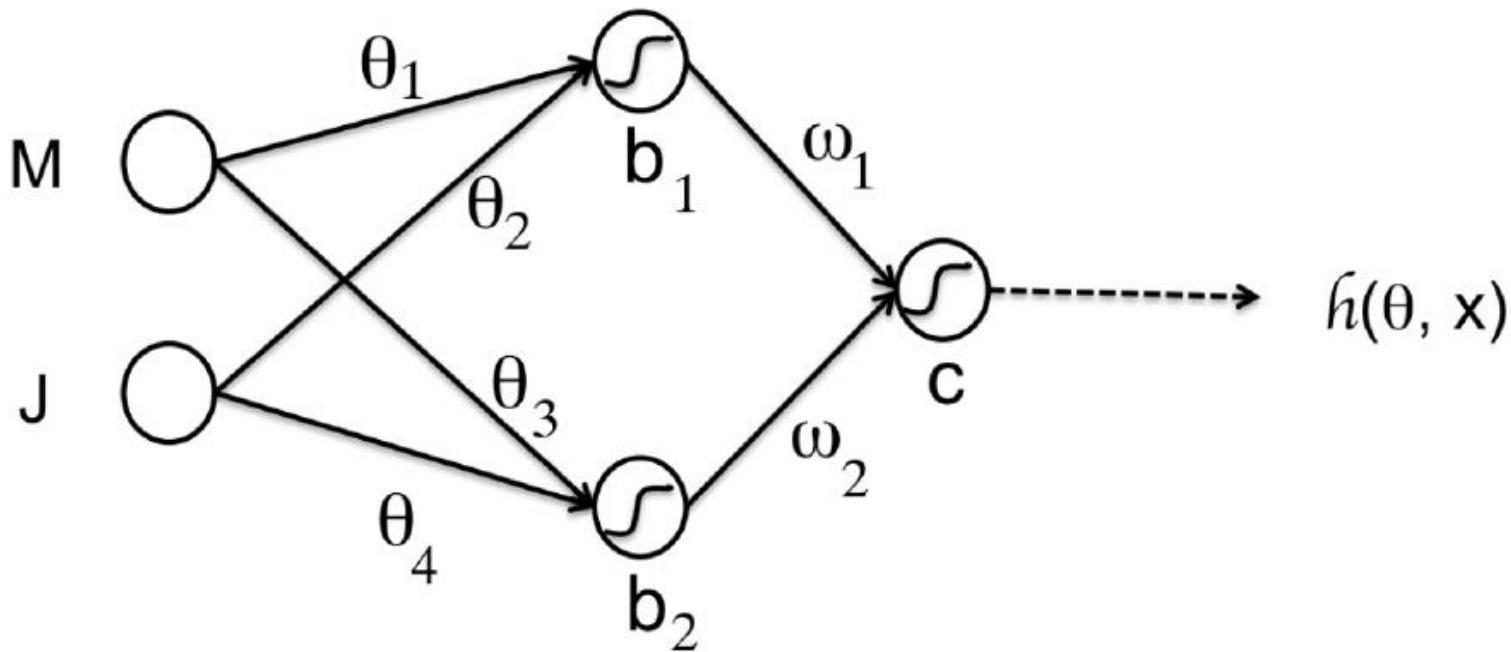




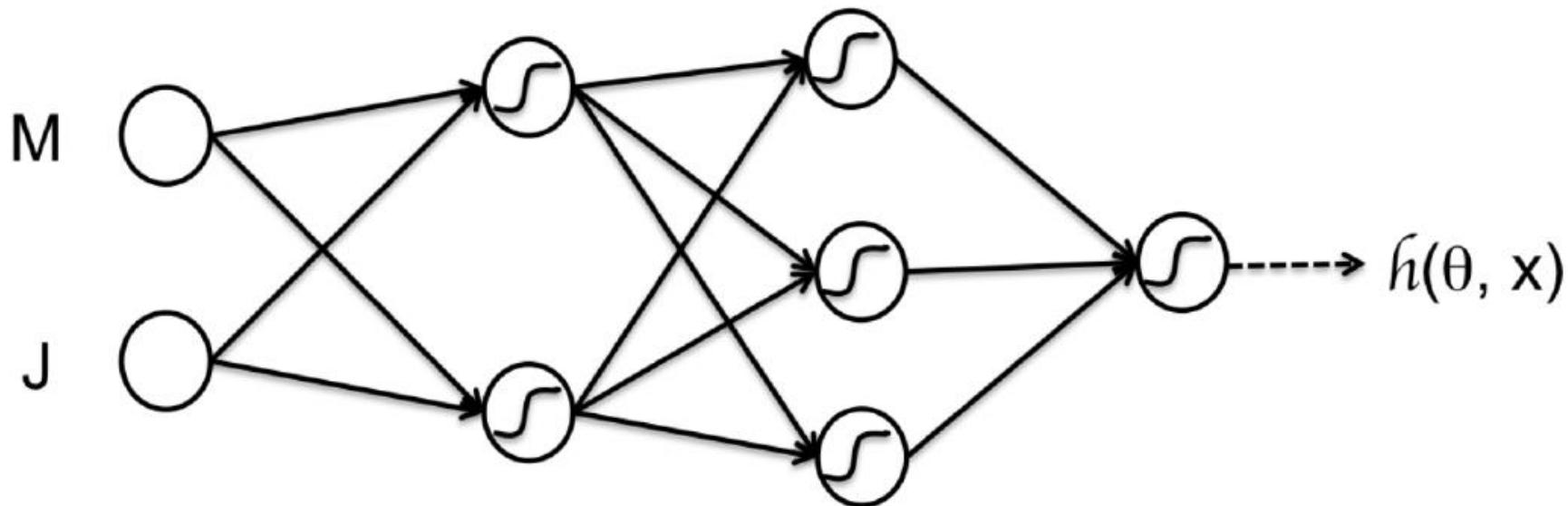
| Movie               | Output by decision function $h_1$ | Output by decision function $h_2$ | Does Nancy like? |
|---------------------|-----------------------------------|-----------------------------------|------------------|
| Lord of the Rings 2 | $h_1(x^{(1)})$                    | $h_2(x^{(1)})$                    | No               |
| ...                 | ...                               | ...                               | ...              |
| Star Wars 1         | $h_1(x^{(n)})$                    | $h_2(x^{(n)})$                    | Yes              |
| Gravity             | $h_1(x^{(n+1)})$                  | $h_2(x^{(n+1)})$                  | ?                |



# THIS IS THE NEURAL NETWORK

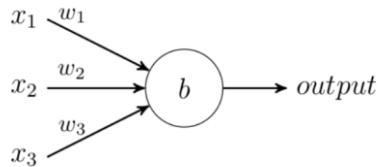


# A DEEPER NEURAL NETWORK



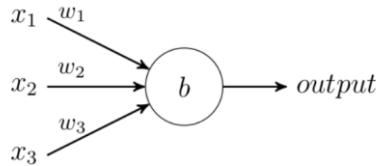
# DEEP LEARNING: BASIC STRUCTURE

## BASIC SINGLE NEURON

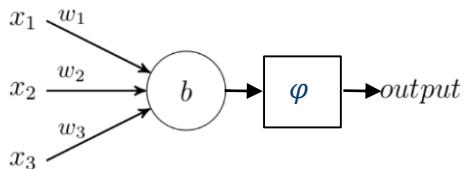


# DEEP LEARNING: BASIC STRUCTURE

BASIC SINGLE NEURON



SINGLE NEURON WITH ACTIVATION

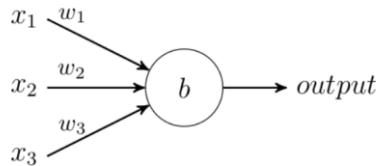


$$u_n = \sum_{j=1}^m w_{nj} x_j$$

$\varphi$  → Activation  
function

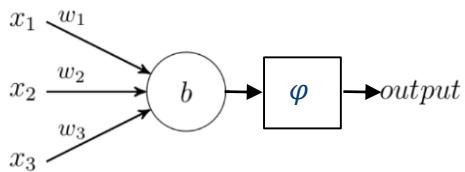
# DEEP LEARNING: BASIC STRUCTURE

BASIC SINGLE NEURON



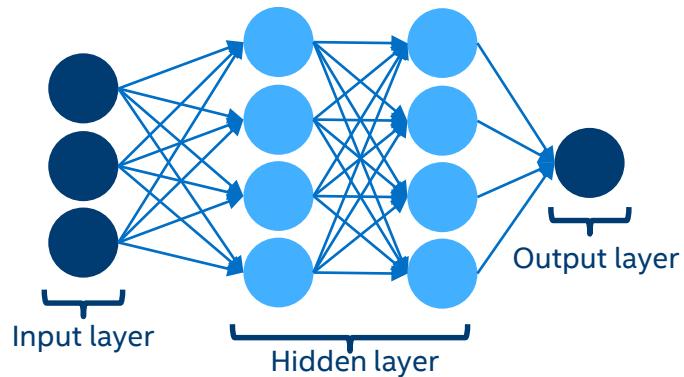
$$u_n = \sum_{j=1}^m w_{nj} x_j$$

SINGLE NEURON WITH ACTIVATION



$\varphi$  → Activation function

BASIC STRUCTURE WITH TWO HIDDEN LAYERS



# KEYWORDS

- Training / Testing percentage
- Overfitting – Underfitting
- Topology
- Training Algorithms
- Learning Rate
- Batch Size

# CLASSICAL MACHINE LEARNING VS DEEP LEARNING

## CLASSIC ML

Using optimized functions or algorithms to extract insights from data

### Algorithms

- Random Forest
- Support Vector Machines
- Regression
- Naïve Bayes
- Hidden Markov
- K-Means Clustering
- Ensemble Methods
- More...

### Inference, Clustering, or Classification

Training Data\*

New Data\*

## DEEP LEARNING

Using massive labeled data sets to train deep (neural) graphs that can make inferences about new data



New Data  
CNN, RNN, RBM...

### Step 1: Training

Hours to Days in Cloud

Use massive labeled dataset (e.g. 10M tagged images) to iteratively adjust weighting of neural network connections



### Step 2: Inference

Real-Time at Edge/Cloud

Form inference about new input data (e.g. a photo) using trained neural network

\*Note: not all classic machine learning functions require training

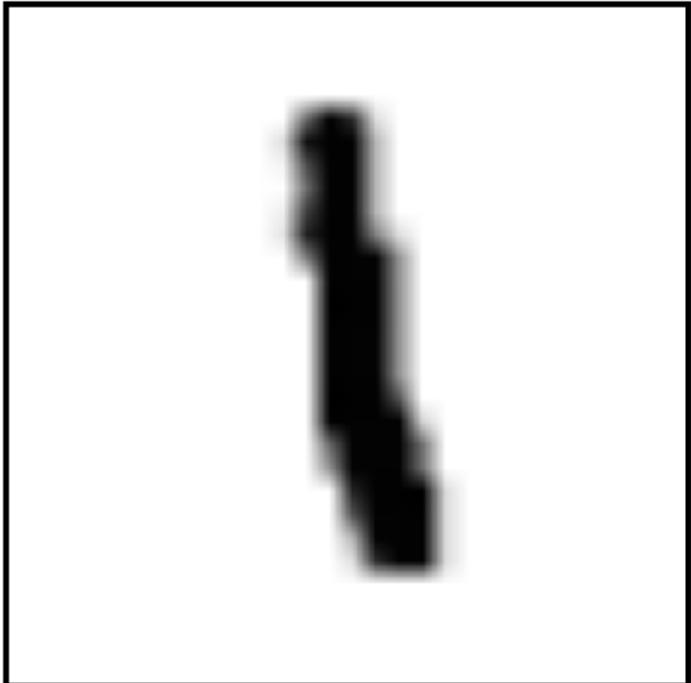


# NAIVE APPROACH

# USE CASE: HANDWRITTEN DIGITS (MNIST)



# USE CASE: HANDWRITTEN DIGITS (MNIST)



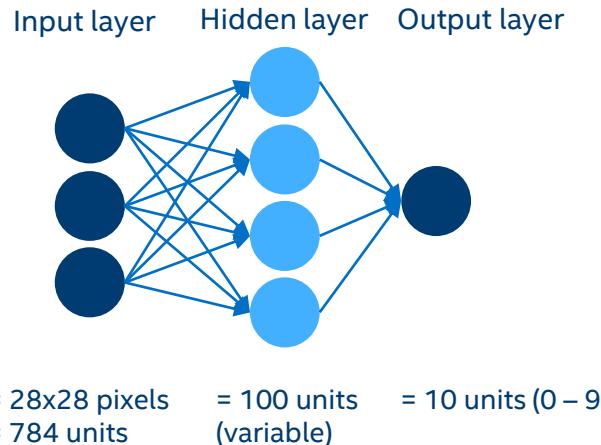
{

|   |   |   |   |   |   |   |    |    |    |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .6 | .8 | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .7 | 1  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .7 | 1  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .5 | 1  | .4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 1  | .4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 1  | .4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 1  | .7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 1  | 1  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .9 | 1  | .1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | .3 | 1  | .1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# USE CASE: HANDWRITTEN DIGITS (MNIST)

|   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| 3 | 4 | 2 | 1 | 9 | 5 | 6 | 2 | 1 | 8 |
| 8 | 9 | 1 | 2 | 5 | 0 | 0 | 6 | 6 | 4 |
| 6 | 7 | 0 | 1 | 6 | 3 | 6 | 3 | 7 | 0 |
| 3 | 7 | 7 | 9 | 4 | 6 | 6 | 1 | 8 | 2 |
| 2 | 9 | 3 | 4 | 3 | 9 | 8 | 7 | 2 | 5 |
| 1 | 5 | 9 | 8 | 3 | 6 | 5 | 7 | 2 | 3 |
| 9 | 3 | 1 | 9 | 1 | 5 | 8 | 0 | 8 | 4 |
| 5 | 6 | 2 | 6 | 8 | 5 | 8 | 8 | 9 | 9 |
| 3 | 7 | 7 | 0 | 9 | 4 | 8 | 5 | 4 | 3 |
| 7 | 9 | 6 | 4 | 7 | 0 | 6 | 9 | 2 | 3 |

MNIST DATASET  
28x28 Pixels



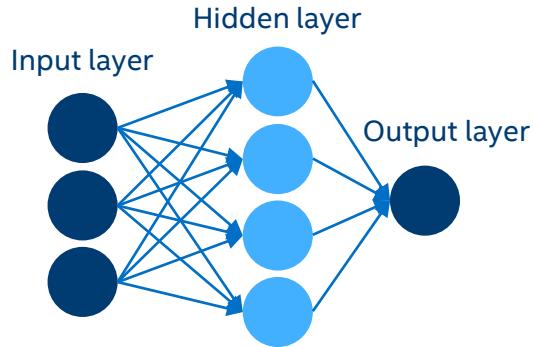
## TOTAL PARAMETERS

|   |           |
|---|-----------|
| $W_{\text{input} \rightarrow \text{hidden}}$  | 784 x 100 |
| $b_{\text{hidden}}$                           | 100       |
| $W_{\text{hidden} \rightarrow \text{output}}$ | 100 x 10  |
| $B_{\text{output}}$                           | 10        |

$$u_n = \sum_{j=1}^m w_{nj} x_j$$

# TRAINING

3



- 1) Initialize weights
- 2) Forward pass
- 3) Calculate cost
- 4) Backward pass
- 5) Update weights

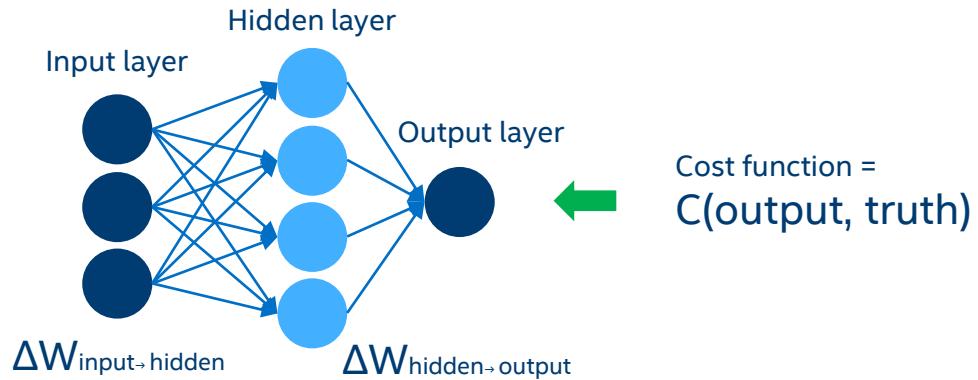
Output    Ground Truth

|     |     |
|-----|-----|
| 0.2 | 0.0 |
| 0.0 | 0.0 |
| 0.5 | 1   |
| 0.0 | 0.0 |
| 0.1 | 0.0 |
| 0.4 | 0.0 |
| 0.2 | 0.0 |
| 0.0 | 0.0 |
| 0.1 | 0.0 |
| 0.0 | 0.0 |

Cost function =  
C(output, truth)

# TRAINING: BACKPROPAGATION

3



# TRAINING: STOCHASTIC GRADIENT DESCENT

3



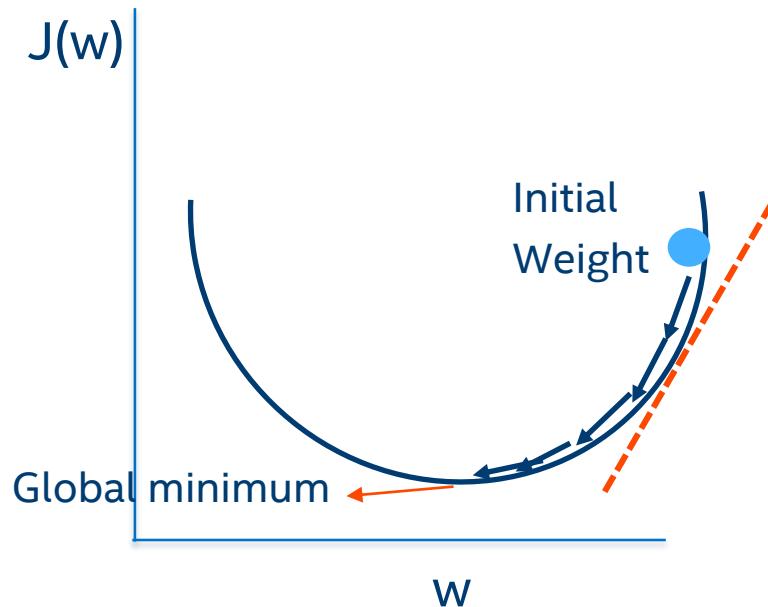
9



4



8





# HANDS-ON WORK

Case Study: MNIST by Softmax Regression

# USE CASE: HANDWRITTEN DIGITS (MNIST)

## Softmax Regression

### iPython notebook:

<https://github.com/mstfldmr/IntelAIWorkshop/blob/master/SoftmaxRegression.ipynb>



# CONVOLUTIONAL NEURAL NETWORKS (CNN)

# Convolutional Neural Networks (CNN)

Essentially neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

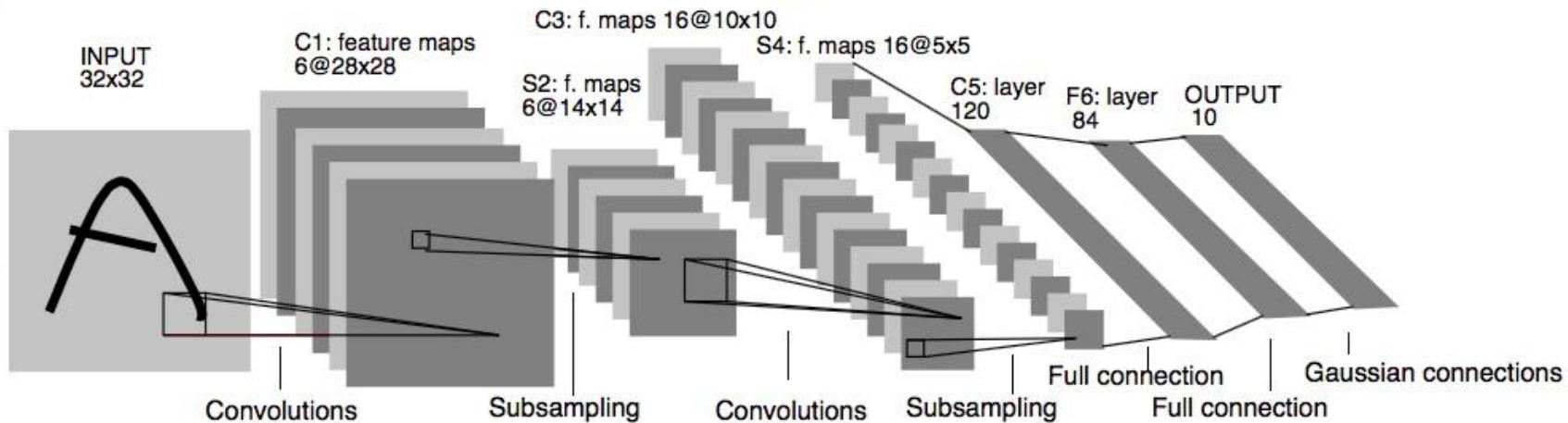
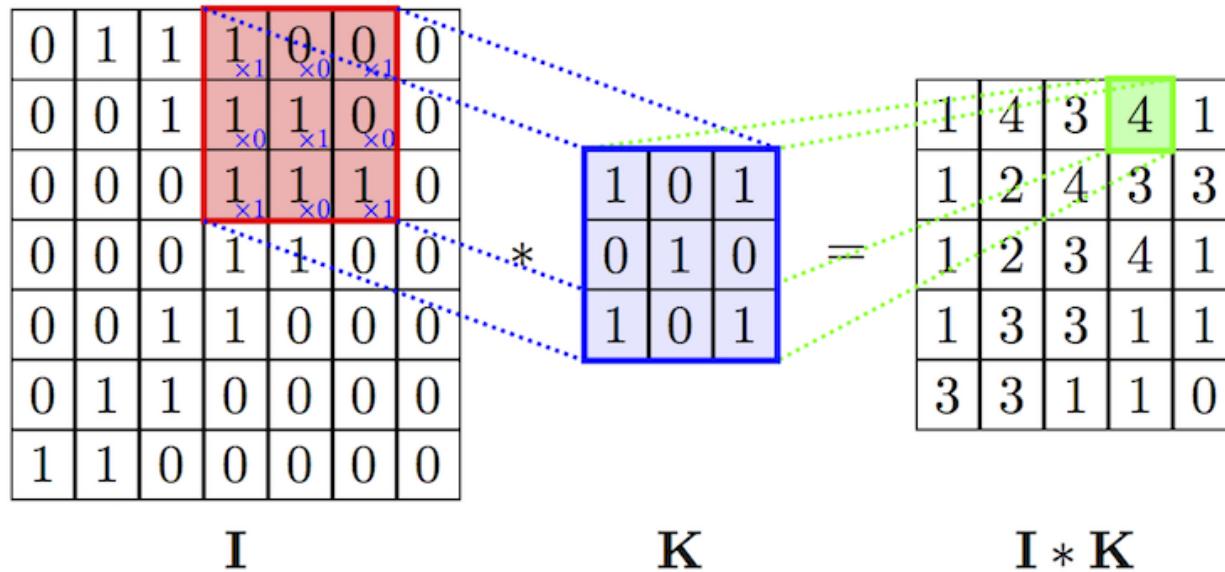


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

# CONVOLUTION



# CONVOLUTION



\*

|   |   |    |
|---|---|----|
| 1 | 0 | -1 |
| 2 | 0 | -2 |
| 1 | 0 | -1 |



# IMAGE KERNELS

identity ▾

$$\left( \begin{array}{ccc} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{array} \right)$$

left sobel ▾

$$\left( \begin{array}{ccc} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{array} \right)$$

bottom sobel ▾

$$\left( \begin{array}{ccc} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{array} \right)$$

sharpen ▾

$$\left( \begin{array}{ccc} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{array} \right)$$

blur ▾

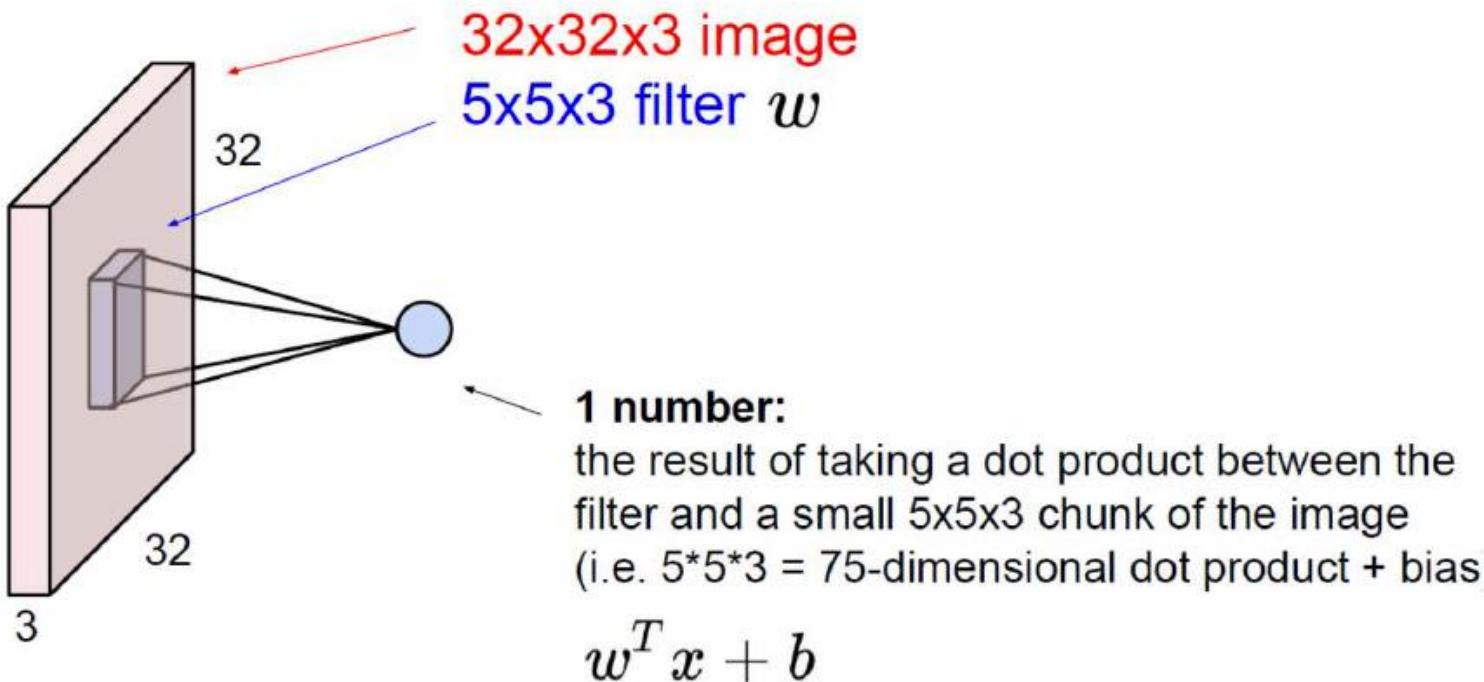
$$\left( \begin{array}{ccc} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{array} \right)$$

<http://setosa.io/ev/image-kernels/>

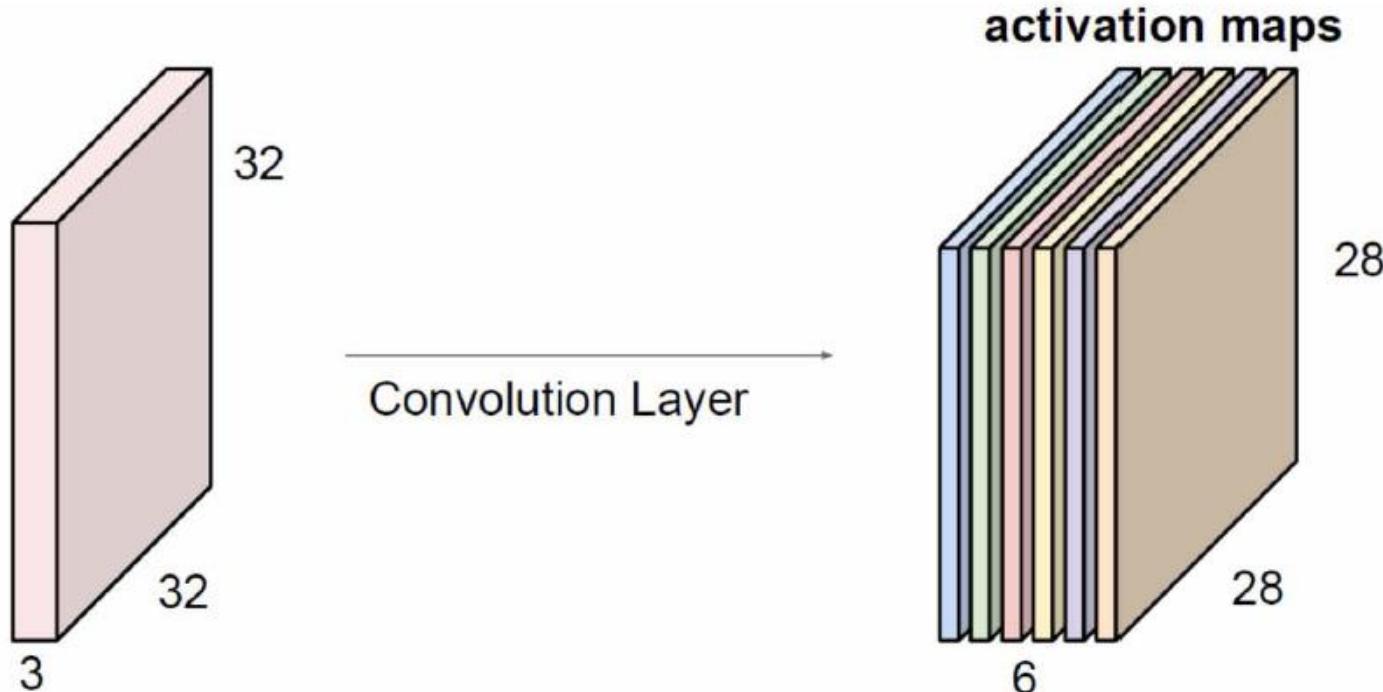
# CONVOLUTION



# Convolution Layer

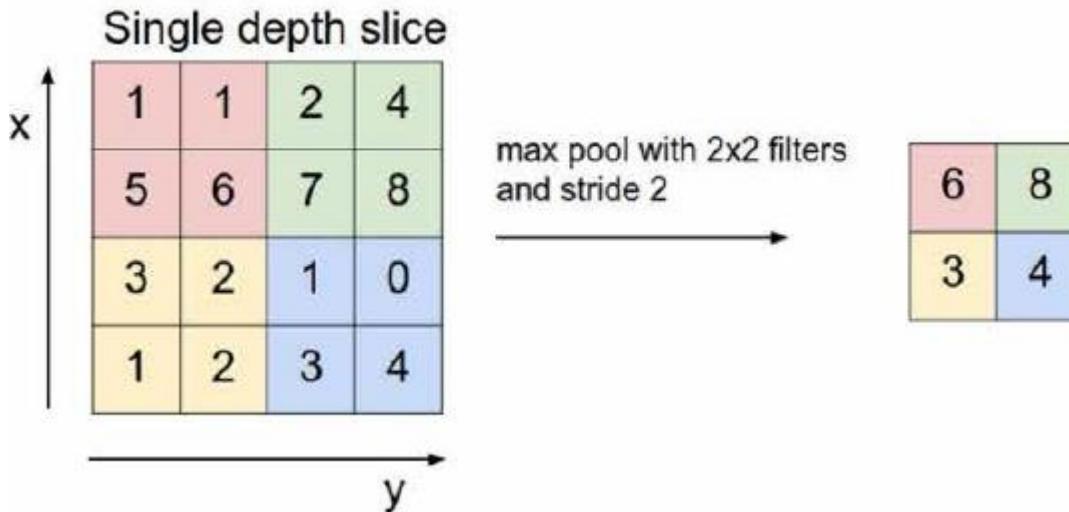


# Convolution Layer

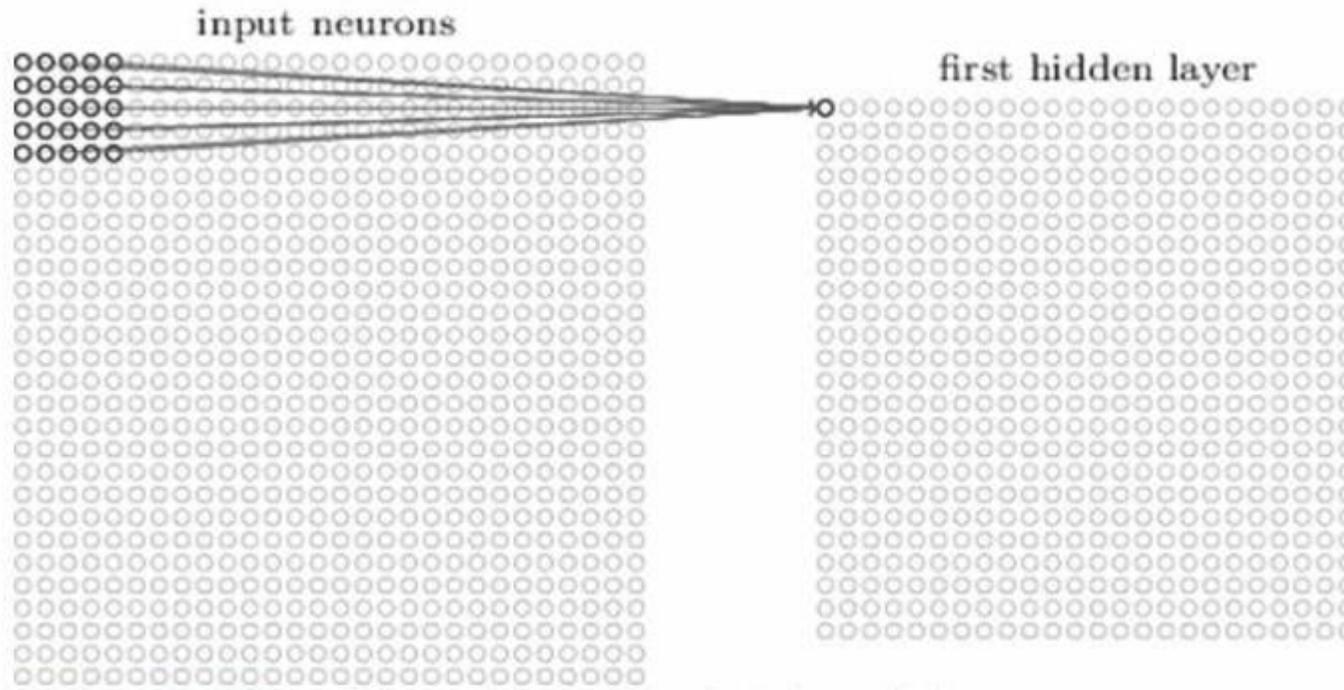


We stack these up to get a “new image” of size  $28 \times 28 \times 6$ !

# Max Pooling

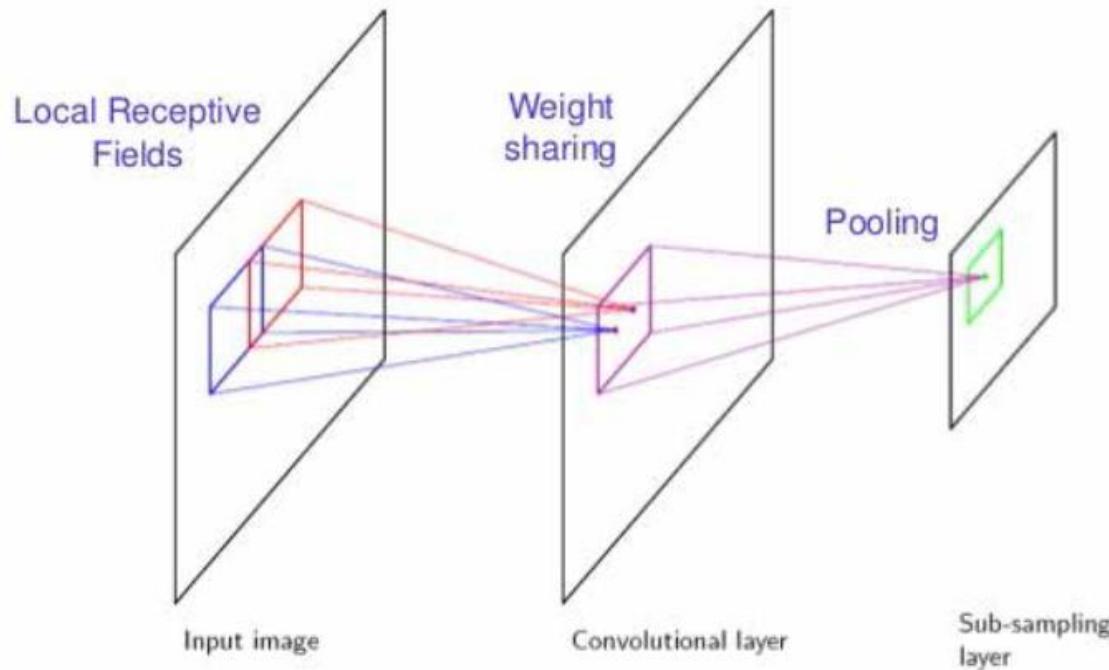


# Simplification 1: Local Receptive Fields

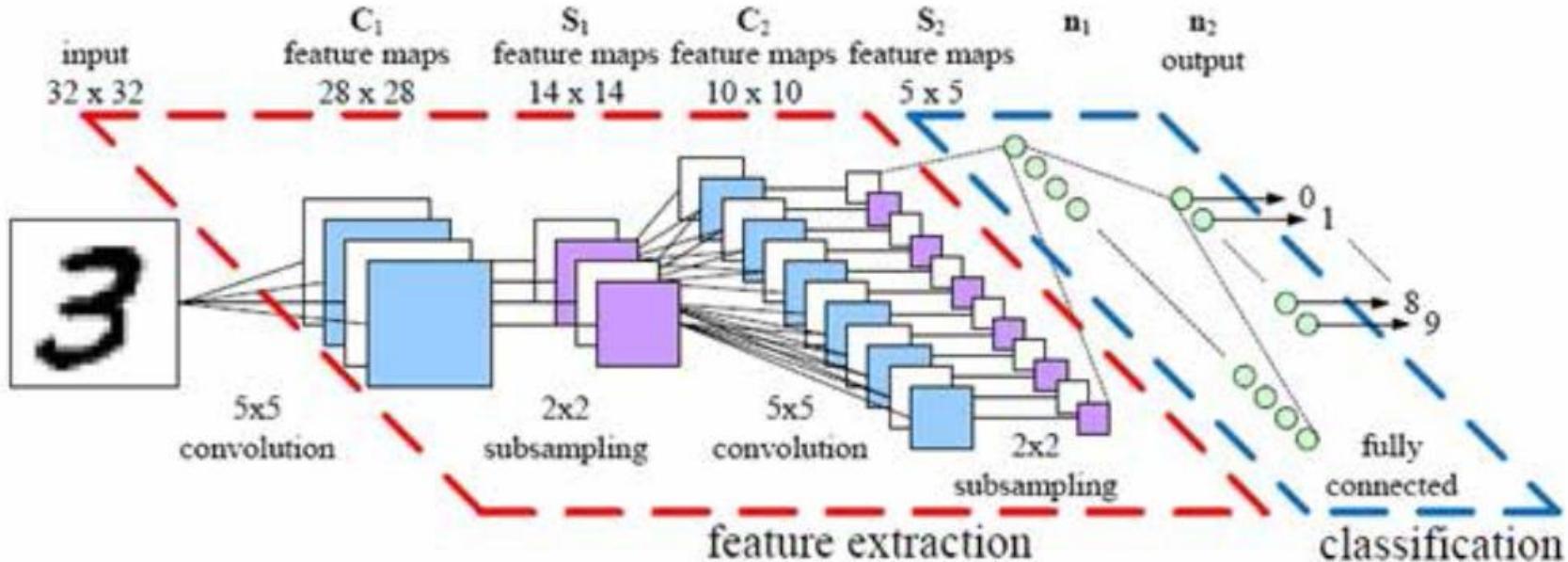


Navigation icons: back, forward, search, etc.

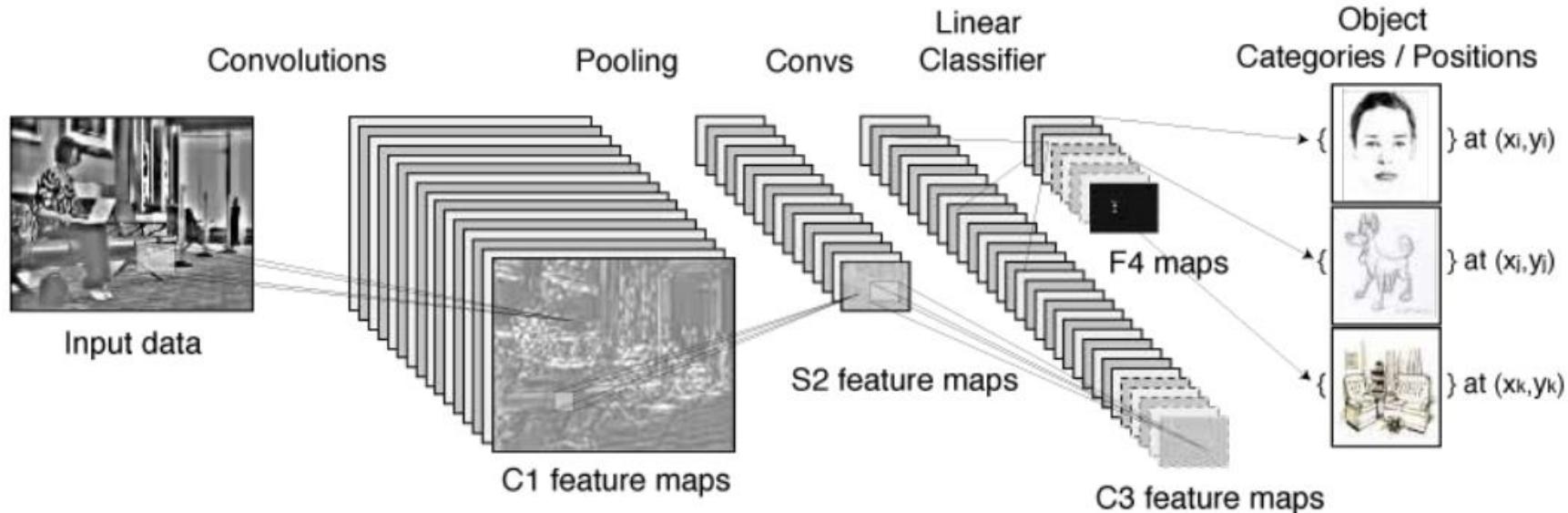
# Simplification 2: Shared Weights



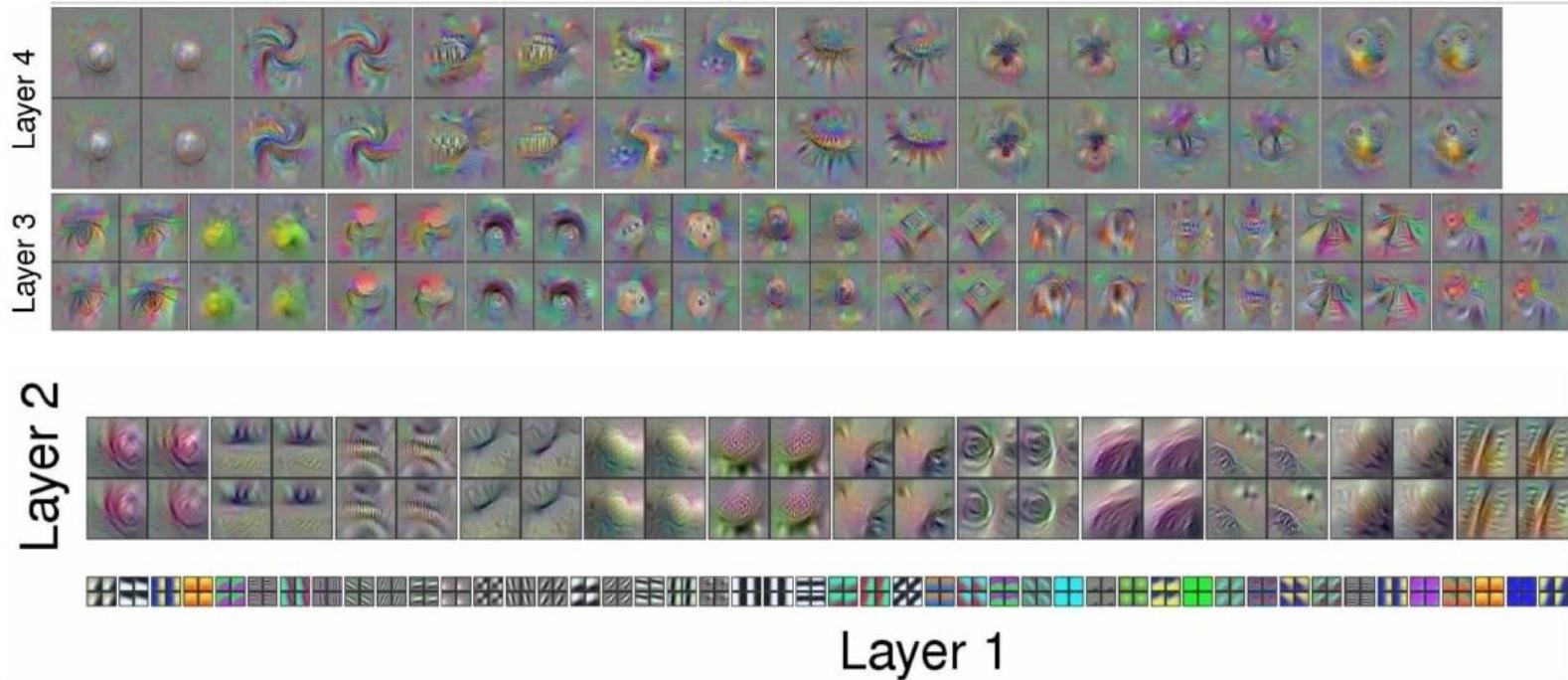
# CNN Pipeline



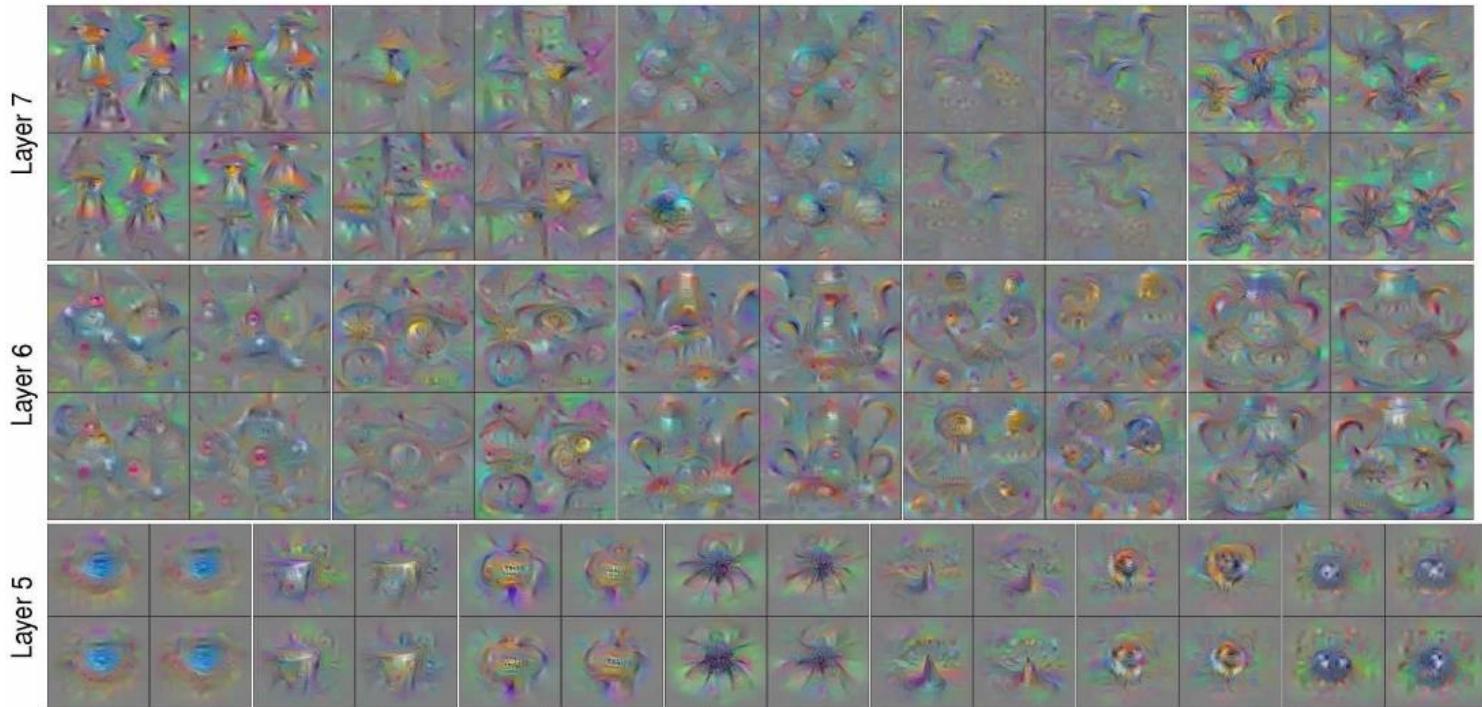
# Example CNN Pipeline for Object Detection



# Visualizing Neurons



# Visualizing Neurons



# Visualizing Neurons

Layer 8



Pirate Ship

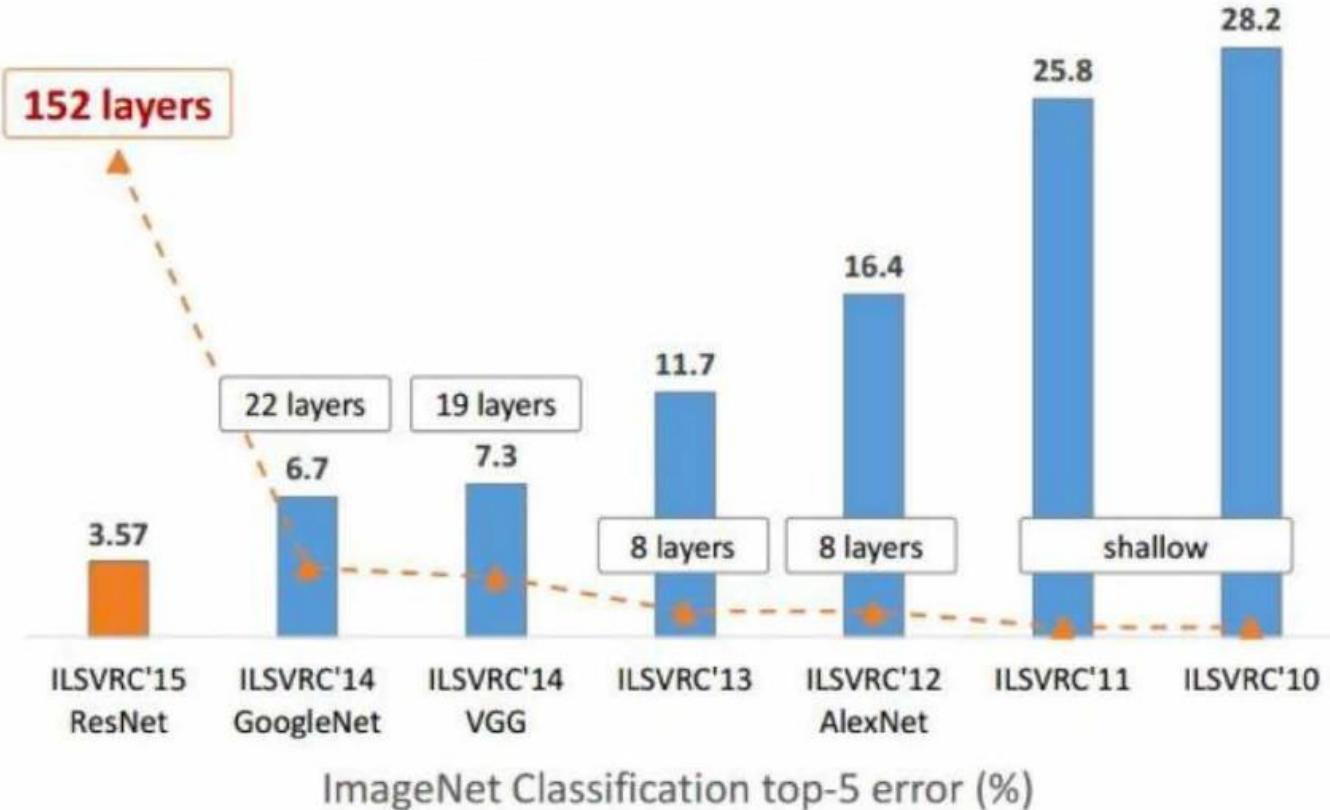
Rocking Chair

Teddy Bear

# ImageNet Large Scale Visual Recognition Competition (ILSVRC)

- ~1M images
- 1K object categories in the training set
- Task: What is the object in the image?
- Classify the image into one of 1000 categories
- Evaluation
  - Is one of the best 5 guesses is correct?
  - Human performance is around 5.1% error.

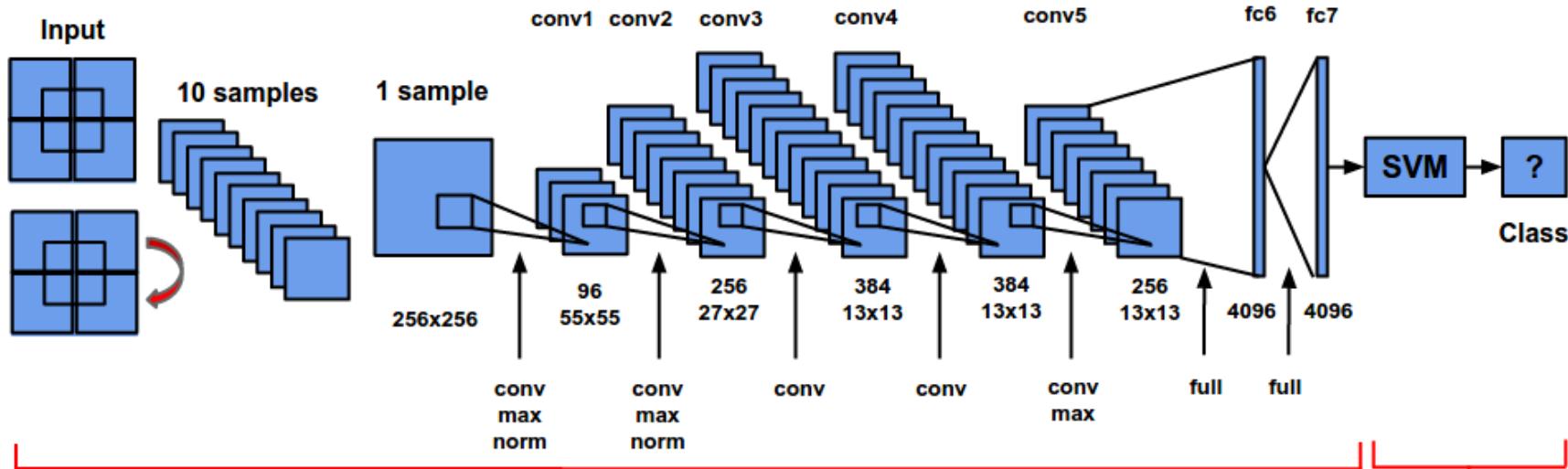
# Evolution of Depth



# AlexNet (2012)

8 layers

16.4 error rate



Extract high level features

Classify each sample

© 2015 Jeremy Karnowski

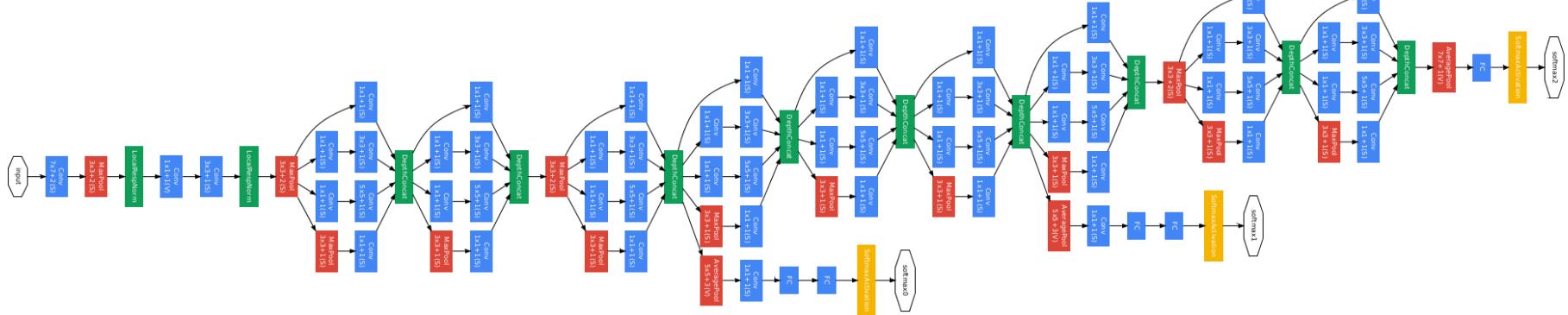


STUDENT DEVELOPER PROGRAM

# GoogleNet (2014)

22 layers

6.7 error rate



# ResNet (2015)

152 layers

3.57 error rate





# HANDS-ON WORK

Case Study: MNIST with Convolutional Neural Networks

# USE CASE: HANDWRITTEN DIGITS (MNIST)

## Convolutional Neural Networks

iPython notebook:

<https://github.com/mstfldmr/IntelAIWorkshop/blob/master/ConvolutionalNeuralNetwork.ipynb>



# HANDS-ON WORK

Case Study: CIFAR10 with Convolutional Neural Networks

# USE CASE: HANDWRITTEN DIGITS (MNIST)

## Convolutional Neural Networks

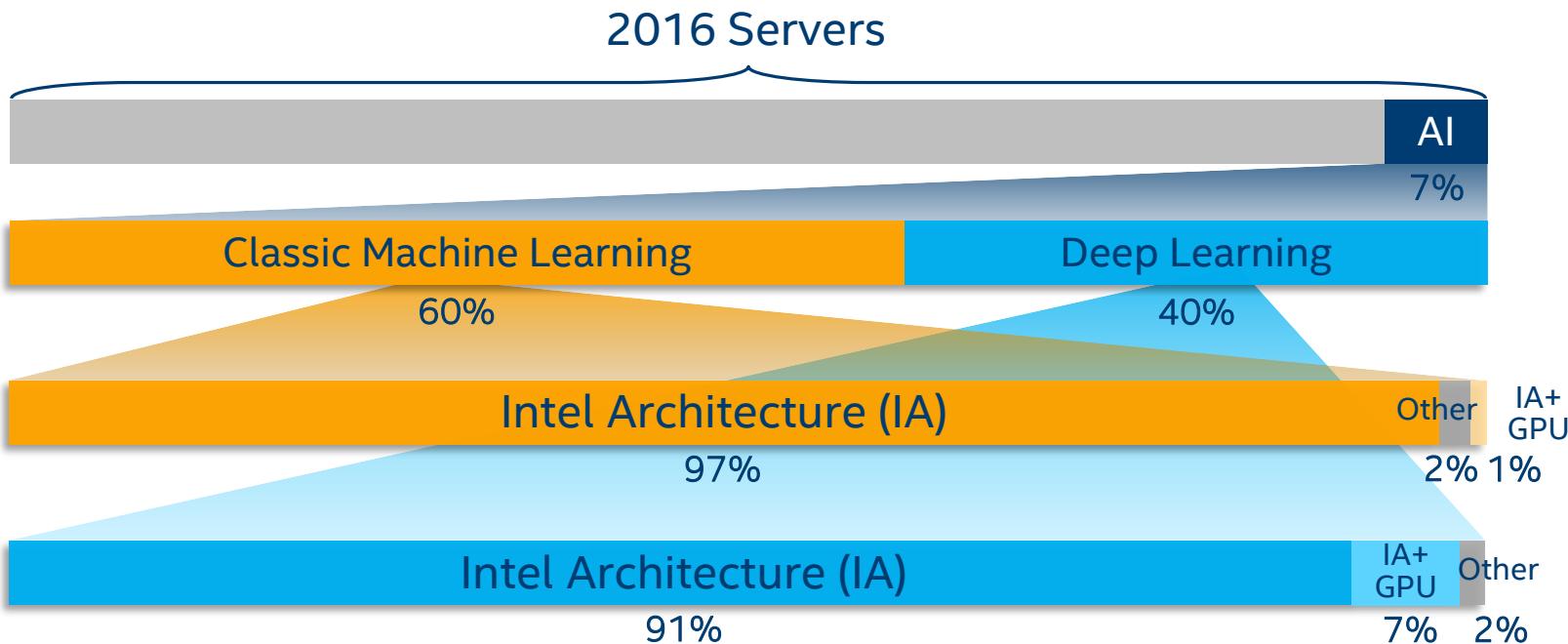
iPython notebook:

<https://raw.githubusercontent.com/mstfldmr/IntelAIWorkshop/master/TFLearn-CIFAR.ipynb>



INTEL® AI PORTFOLIO

# AI IS THE FASTEST GROWING DATA CENTER WORKLOAD



# INTEL® NERVANA™ AI ACADEMY

Hone Your Skills and Build the Future of AI



TRAINING



TOOLS



COMMUNITY

Benefit from expert-led trainings, hands-on workshops, exclusive remote access, and more.

Gain access to the latest libraries, frameworks, tools and technologies from Intel to accelerate your AI project.

Collaborate with industry luminaries, developers, students, and Intel engineers.

[software.intel.com/ai/academy](http://software.intel.com/ai/academy)

# INTEL AI PORTFOLIO

## EXPERIENCES



## TOOLS



Intel® Deep  
Learning SDK

Intel® Computer  
Vision SDK

Movidius Neural  
Compute Stick

saffron  
TECHNOLOGY  
an Intel company

## FRAMEWORKS



theano



Caffe



E2E Tool

## LIBRARIES



Intel® DAAL

Intel® Nervana™ Graph\*  
Intel® MKL MKL-DNN Intel® MLSL

Movidius  
MvTensor  
Library

Associative  
Memory Base

## HARDWARE



Compute



Memory & Storage



Networking



Visual Intelligence

UNLEASH  
FULL  
POTENTIAL

\*Coming 2017

# LIBRARIES, FRAMEWORKS & TOOLS

|                            | <b>Intel® Math Kernel Library</b><br> Intel® MKL | <b>MKL-DNN</b><br>                | <b>Intel® MLSL</b><br>    | <b>Intel® Data Analytics Acceleration Library (DAAL)</b><br> | <b>Distribution</b><br> | <b>Open Source Frameworks</b><br> | <b>Intel Deep Learning SDK</b><br> | <b>Intel® Computer Vision SDK</b><br>      |
|----------------------------|---|--|---|---|--|--|---|---|
| <b>High Level Overview</b> | Computation primitives; high performance math primitives granting low level of control  | Computation primitives; free open source DNN functions for high-velocity integration with deep learning frameworks | Communication primitives; building blocks to scale deep learning framework performance over a cluster       | Broad data analytics acceleration object oriented library supporting distributed ML at the algorithm level                                      | Most popular and fastest growing language for machine learning   | Toolkits driven by academia and industry for training machine learning algorithms                                    | Accelerate deep learning model design, training and deployment  | Toolkit to develop & deploying vision-oriented solutions that harness the full performance of Intel CPUs and SOC accelerators |
| <b>Primary Audience</b>    | Consumed by developers of higher level libraries and Applications   | Consumed by developers of the next generation of deep learning frameworks  | Deep learning framework developers and optimizers   | Wider Data Analytics and ML audience, Algorithm level development for all stages of data analytics  | Application Developers and Data Scientists   | Machine Learning App Developers, Researchers and Data Scientists.  | Application Developers and Data Scientists  | Developers who create vision-oriented solutions   |
| <b>Example Usage</b>       | Framework developers call matrix multiplication, convolution functions  | New framework with functions developers call for max CPU performance   | Framework developer calls functions to distribute Caffe training compute across an Intel® Xeon Phi™ cluster | Call distributed alternating least squares algorithm for a recommendation system  | Call scikit-learn k-means function for credit card fraud detection   | Script and train a convolution neural network for image recognition  | Deep Learning training and model creation, with optimization for deployment on constrained end device                 | Use deep learning to do pedestrian detection  |

Find out more at <http://software.intel.com/ai>

# INTEL DISTRIBUTION FOR PYTHON

Advancing Python Performance Closer to Native Speeds



For developers using the most popular and fastest growing programming language for AI

## Easy, Out-of-the-box Access to High Performance Python

- Prebuilt, optimized for numerical computing, data analytics, HPC
- Drop in replacement for your existing Python (no code changes required)

## Drive Performance with Multiple Optimization Techniques

- Accelerated NumPy/SciPy/Scikit-Learn with Intel® MKL
- Data analytics with pyDAAL, enhanced thread scheduling with TBB, Jupyter\* Notebook interface, Numba, Cython
- Scale easily with optimized MPI4Py and Jupyter notebooks

## Faster Access to Latest Optimizations for Intel Architecture

- Distribution and individual optimized packages available through conda and Anaconda Cloud
- Optimizations upstreamed back to main Python trunk

[software.intel.com/intel-distribution-for-python](http://software.intel.com/intel-distribution-for-python)



## Math Kernel Library for Deep Neural Networks

For developers of deep learning frameworks featuring optimized performance on Intel hardware

### Distribution Details

- Open Source
- Apache 2.0 License
- Common DNN APIs across all Intel hardware.
- Rapid release cycles, iterated with the DL community, to best support industry framework integration.
- Highly vectorized & threaded for maximal performance, based on the popular Intel® MKL library.

BETA Now Available!

[github.com/01org/mkl-dnn](https://github.com/01org/mkl-dnn)

Direct 2D Convolution

Local response normalization (LRN)

Rectified linear unit neuron activation (ReLU)

Maximum pooling

Inner product

# INTEL® MACHINE LEARNING SCALING LIBRARY (MLSL)

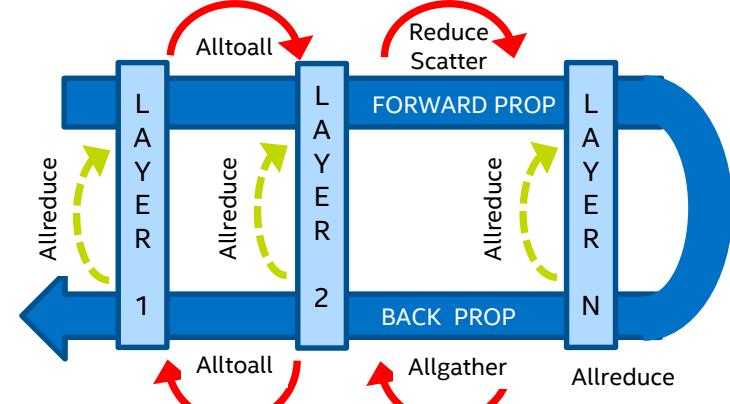
Scaling Deep Learning to 32 Nodes and Beyond

For maximum deep learning scale-out performance on Intel® architecture



## Deep learning abstraction of message-passing implementation

- Built on top of MPI; allows other communication libraries to be used as well
- Optimized to drive scalability of communication patterns
- Works across various interconnects: Intel® Omni-Path Architecture, InfiniBand, and Ethernet
- Common API to support Deep Learning frameworks (Caffe, Theano, Torch etc.)



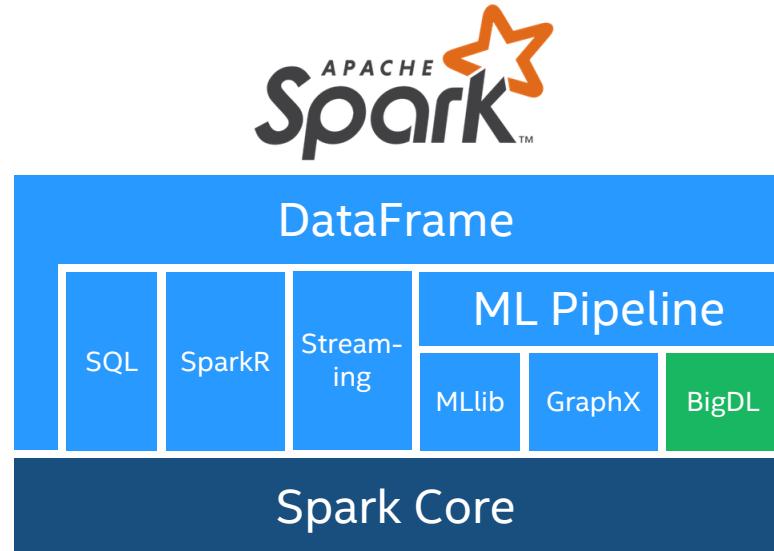
[github.com/01org/MLSL/releases](https://github.com/01org/MLSL/releases)



## Bringing Deep Learning to Big Data

For developers looking to run deep learning on Hadoop/Spark due to familiarity or analytics use

- **Open Sourced** Deep Learning Library for Apache Spark\*
- **Make Deep learning more Accessible** to Big data users and data scientists.
- **Feature Parity** with popular DL frameworks like Caffe, Torch, Tensorflow etc.
- **Easy Customer and Developer Experience**
  - Run Deep learning Applications as Standard Spark programs;
  - Run on top of existing Spark/Hadoop clusters (No Cluster change)
- **High Performance** powered by Intel MKL and Multi-threaded programming.
- **Efficient Scale out** leveraging Spark architecture.



[github.com/intel-analytics/BigDL](https://github.com/intel-analytics/BigDL)



# NEXT STEPS....

# INTEL® NERVANA™ AI ACADEMY

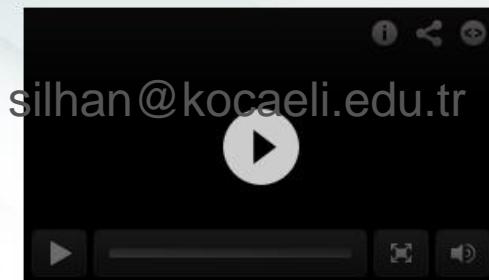
Sharpen your machine learning skills and create the future of artificial intelligence

## Getting Started



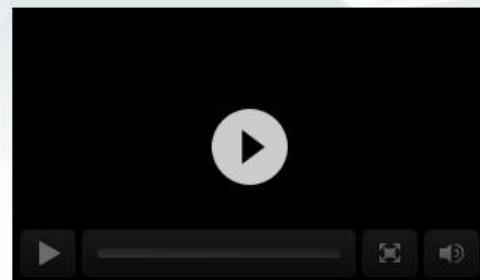
### Machine Learning 101

Rely on the Intel® Nervana™ AI Academy to help you increase your knowledge base and



### Deep Learning 101

In this webinar, we describe various deep learning uses and highlight those in which



### Deep Learning 102: Neural Networks, Cost Functions, and More

<https://software.intel.com/ai/academy>

# ONLINE COURSES

- Stanford Convolutional Neural Networks  
for Visual Recognition: <http://cs231n.github.io>
- MIT Machine Learning and Neural Networks:  
<http://www.ai.mit.edu/courses/6.891-f99>
- MIT Self Driving Cars: <https://selfdrivingcars.mit.edu>
- Deep Learning AI: <https://www.deeplearning.ai>
- Coursera: <https://www.coursera.org>
- Udacity: <https://www.udacity.com>
- Udemy: <https://www.udemy.com>
- ...

# TURKISH CONTENT

- Deep Learning Türkiye:

<https://github.com/deeplearningturkiye/turkce-yapay-zeka-kaynaklari>



# Competitions

13 active competitions

Sort by Prize ▾

Active All Entered

All Categories ▾

Search



## Zillow Prize: Zillow's Home Value Prediction (Zestimate)

Can you improve the algorithm that changed the world of real estate?

Featured - 7 months to go

\$1,200,000

847 teams



## Intel & MobileODT Cervical Cancer Screening

Which cancer treatment will be most effective?

Featured - 6 days to go

\$100,000

848 teams



## Planet: Understanding the Amazon from Space

Use satellite data to track the human footprint in the Amazon rainforest

\$60,000

469 teams

<https://www.kaggle.com>



# Q&A



# STUDENT DEVELOPER PROGRAM