Resources

Books

- *“Deep Learning”* by Ian Goodfellow, Yoshua Bengio and Aaron Courville (MIT Press, 2016)
- *“Neural Networks and Deep Learning”* by Michael Nielsen
- [classical CV] *"Computer Vision: Models, Learning, and Inference”* by Simon J.D. Prince

Courses

- Stanford CS 231n: by Li, Karpathy & Johnson
  http://cs231n.github.io/
- fast.ai online courses (all are free and have no ads)
  https://www.fast.ai/

Goals of the Course

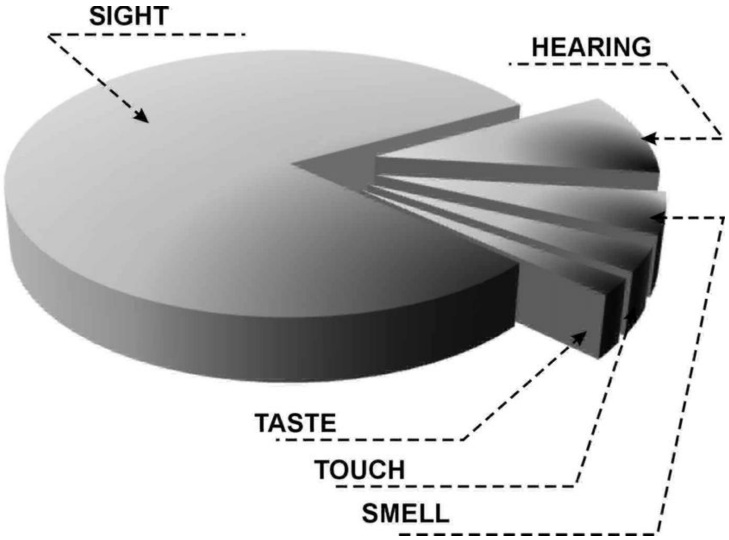
- Get an overview of Convolutional Neural Networks (CNN) with application to Computer Vision;
- Get understanding of various elements/blocks of CNNs and typical network topologies;
- Get deeper with one particular problem (Object Detection);

- Device and train CNN for classification
- Learn useful tools: Pytorch, Colab notebook, TensorBroad

Content of today lecture

- **Intro**
- ML review
- Intro to CNN

Intro



The goal of computer vision is to extract useful information from visual input (images, video)

Intro

- indoor/outdoor? [image classification]
- Where are the objects? [object detection]
- How far is the object ? [depth estimation]
- What people are doing? [activity recognition]
- Is the state of the environment normal? [anomaly detection]
- ...

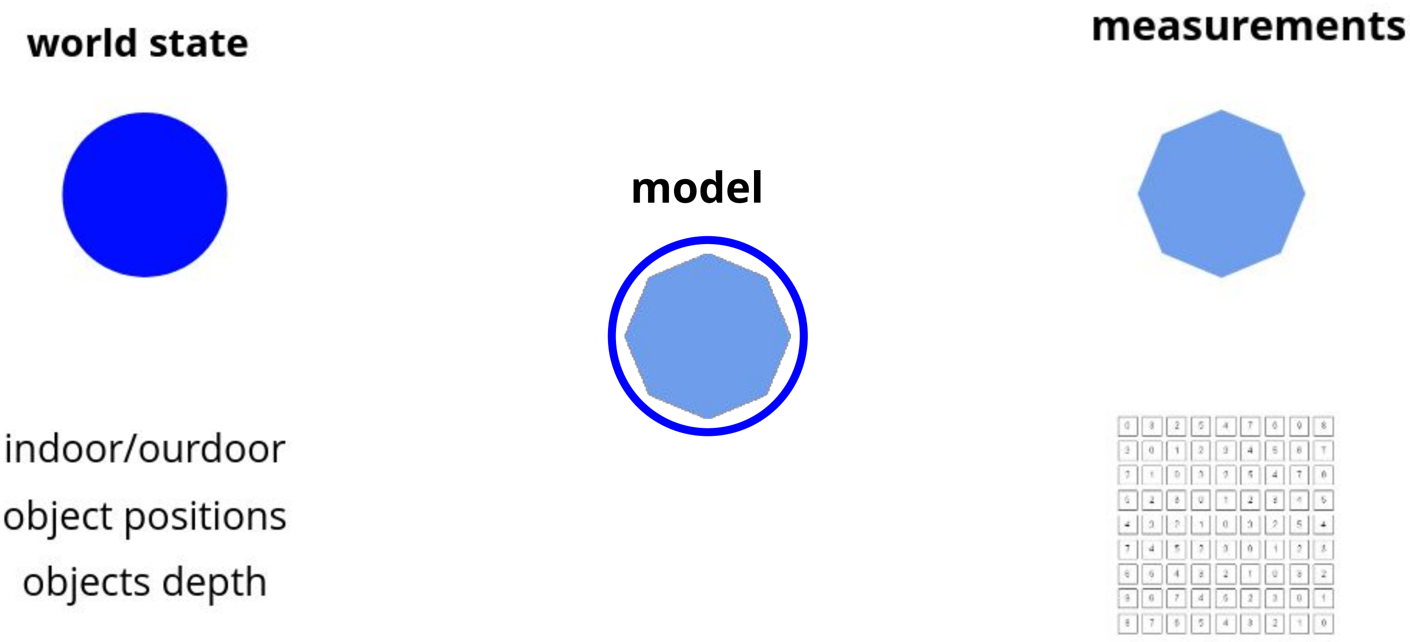
Intro

what humans see

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

what computers see

Intro

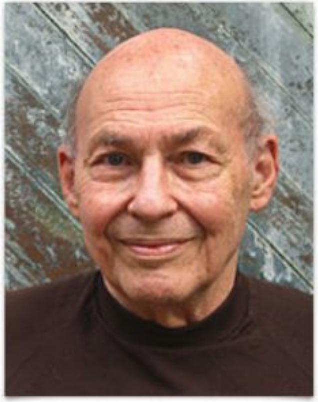


“The vision problem (or goal) is to use the measurements to infer the world state”

Simon J.D. Prince "Computer Vision: Models, Learning, and Inference"

Intro

- **1966:** *Marvin Minsky* posed the development of a computer vision system as an undergraduate summer project.

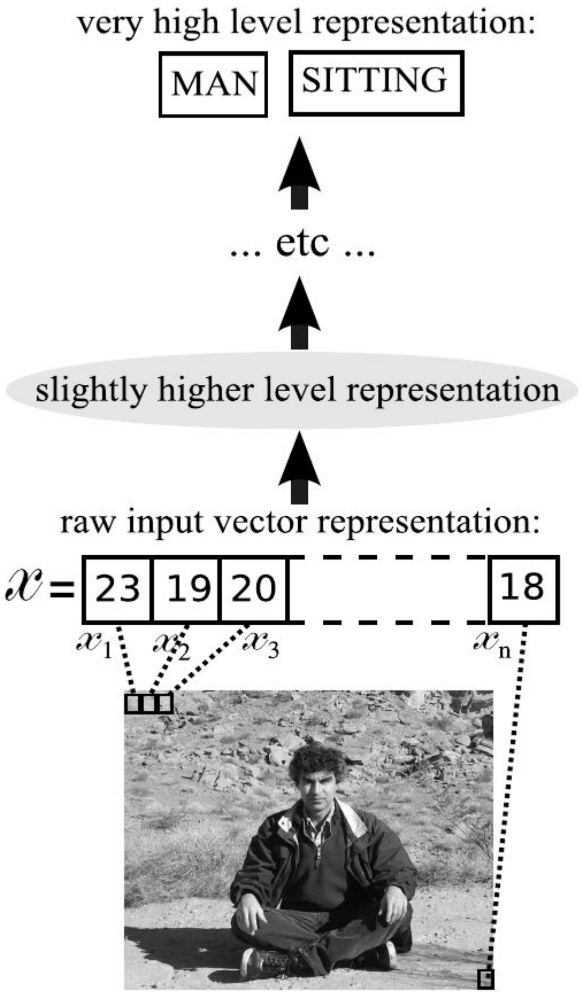


- **1970's:** some progress on interpreting selected images
- **1980's:** NNs come and go; shift toward geometry and increased mathematical rigor
- **1990's:** face recognition;
- **2000's:** broader recognition; large annotated datasets available; video processing
- **2010's:** DNN, convolutional NN

Intro

Visual scene is hierarchically organized

object	trees
↑	↑
object parts	bark, leaves, etc.
↑	↑
primitive features	oriented edges
↑	↑
input image	forest image



https://ttic.uchicago.edu/~shubhendu/Pages/CMSC35246.html

Intro

Visual scene is hierarchically organized

object
↑
object parts
↑
primitive features
↑
input image

trees
↑
bark, leaves, etc.
↑
oriented edges
↑
forest image

Inferotemporal cortex

V4: different textures

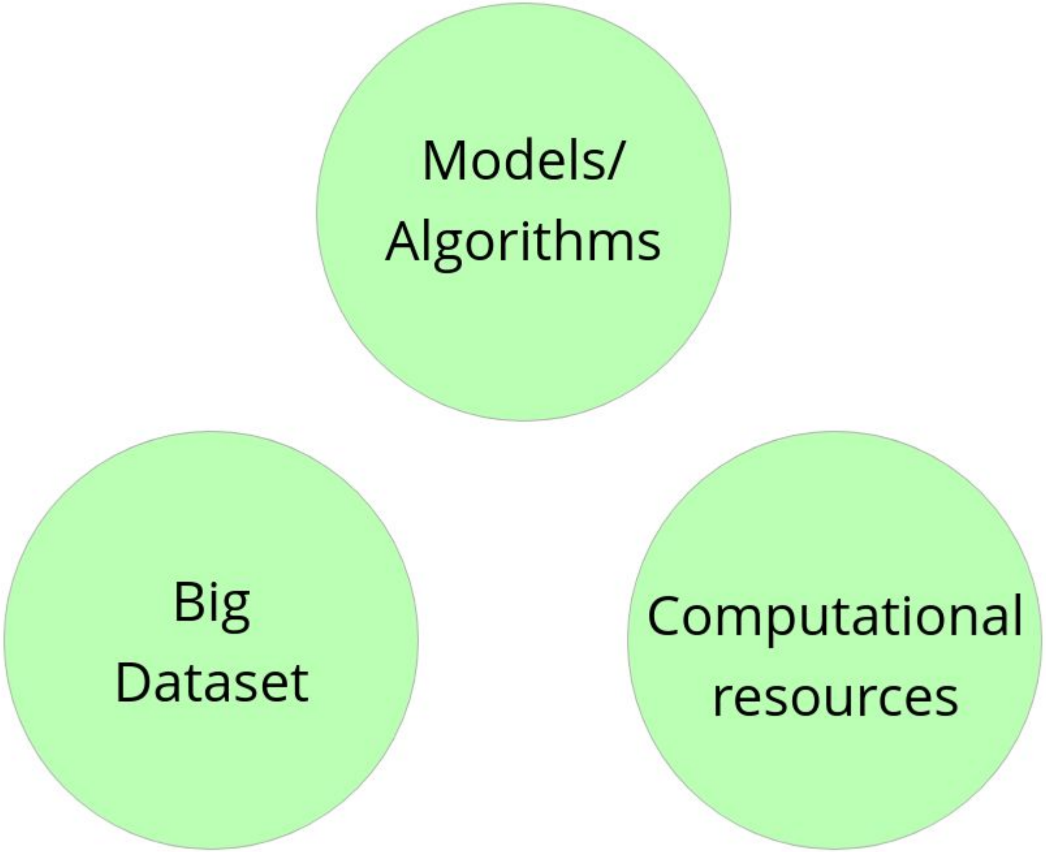
V1: simple and complex cells

photo-receptors retina

Biological vision is hierarchically organized too!

https://ttic.uchicago.edu/~shubhendu/Pages/CMSC35246.html

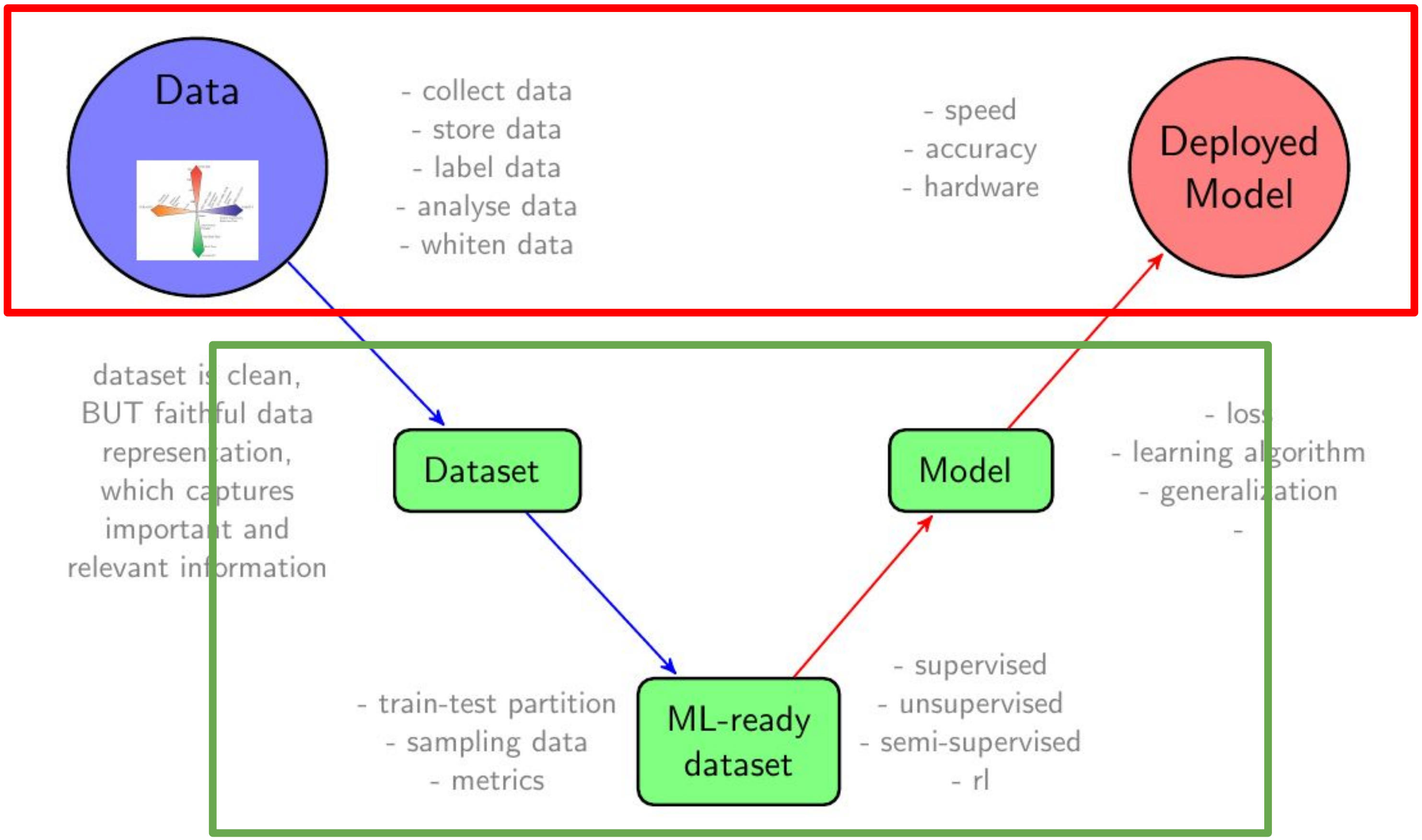
Intro



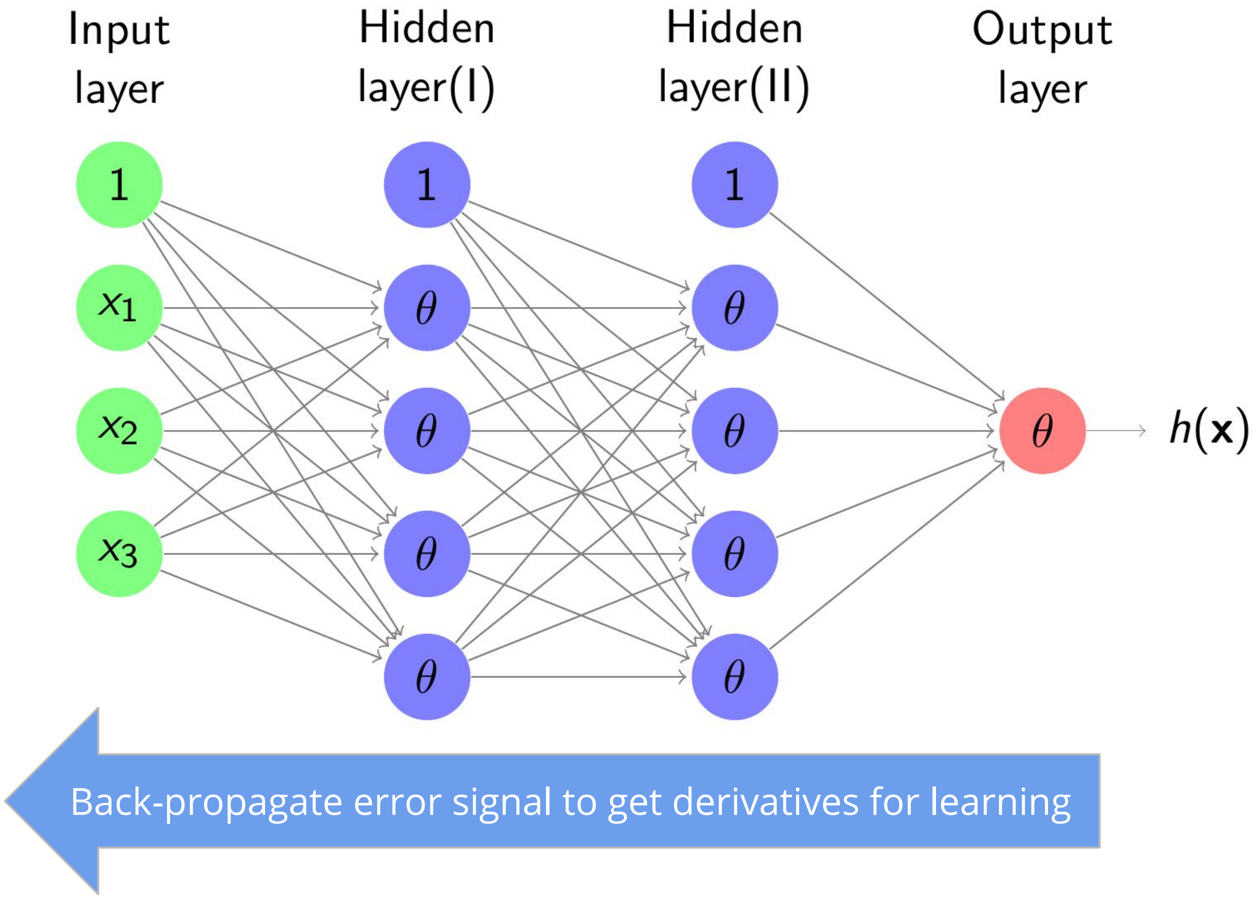
Content

- Intro
- **ML review**
- Intro to CNN

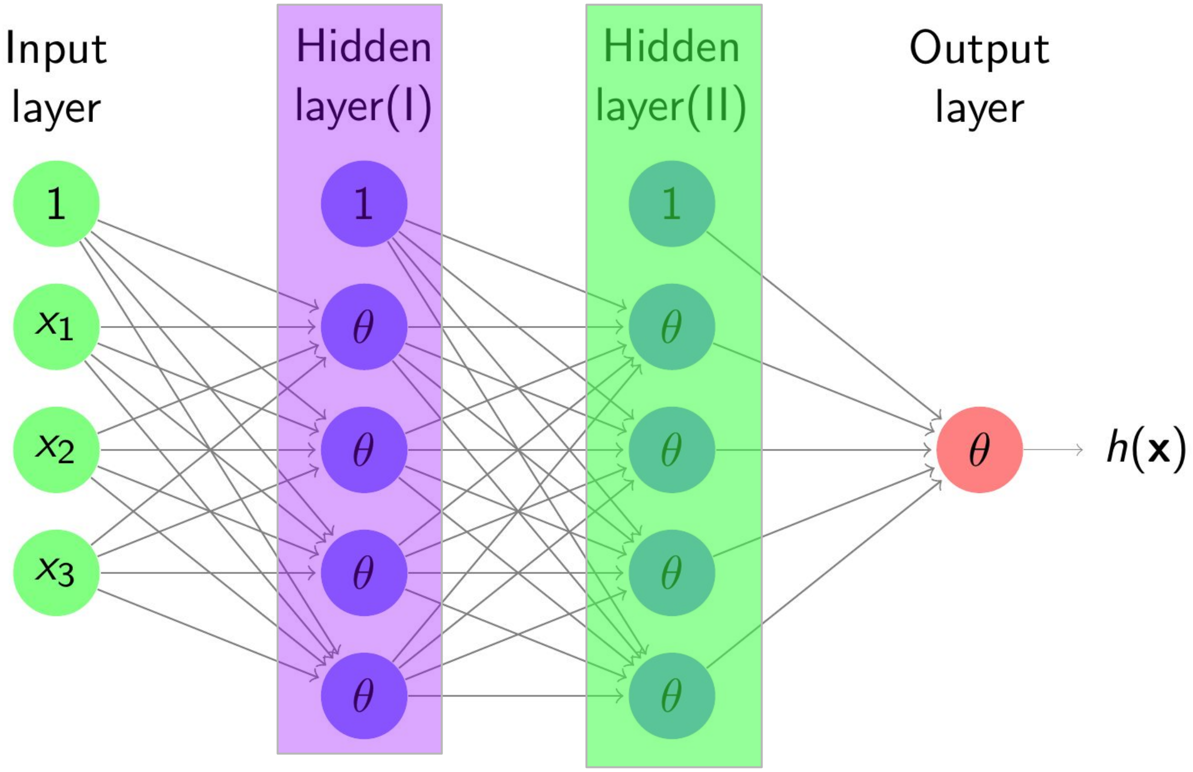
ML review


Data
- collect data
- store data
- label data
- analyse data
- whiten data
- speed
- accuracy
- hardware
Deployed Model
dataset is clean,
BUT faithful data
representation,
which captures
important and
relevant information
Dataset
Model
- loss
- learning algorithm
- generalization
-
- train-test partition
- sampling data
- metrics
ML-ready dataset
- supervised
- unsupervised
- semi-supervised
- rl

ML review (NN)


Input layer
Hidden layer(I)
Hidden layer(II)
Output layer
1
x_1
x_2
x_3
1
θ
1
θ
θ
$h(\mathbf{x})$
Back-propagate error signal to get derivatives for learning

ML review (NN)

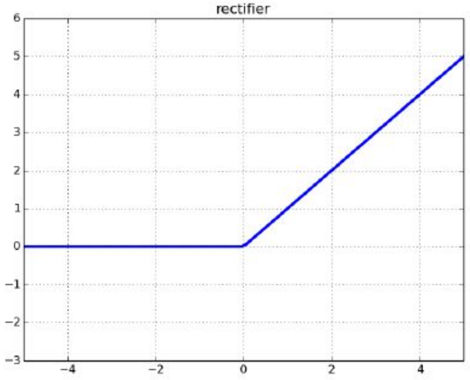


$$\{f(\boldsymbol{x};\boldsymbol{\theta}) = \mathbf{W}_L\boldsymbol{\sigma}_L(\mathbf{W}_{L-1}\cdots\boldsymbol{\sigma}_2(\mathbf{W}_2\boldsymbol{\sigma}_1(\mathbf{W}_1\boldsymbol{x}))) \mid \boldsymbol{\theta} = \{\mathbf{W}_1,\ldots,\mathbf{W}_L\}\}$$

ML review (NN)

parameters in NN:

$$W_l^{ij} = \begin{cases} 1 \le l \le L & \text{layers} \\ 0 \le i \le d^{(l-1)} & \text{inputs} \\ 1 \le j \le d^{(l)} & \text{outputs} \end{cases}$$



activation:

$$x_j^{(l)} = \sigma(s_j^{(l)}) = \sigma\Big(\sum_{i=0}^{d^{(l-1)}} W_l^{ij} x_i^{(l-1)}\Big)$$

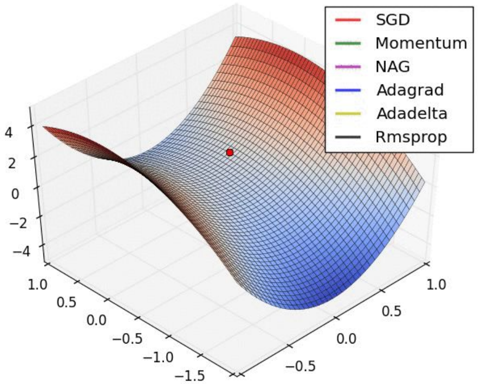
$$\sigma(s) = RELU(s) = max(0, x)$$

ML review (NN)

Define **Loss (or Cost) function**:

$$L_{logloss} = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{M} y_{nk} \cdot log(p_{nk})$$

$$L_{rmse} = \frac{1}{N}\sum_{n=1}^{N} \parallel F(\mathbf{x_n}) - y_i \parallel_2$$



Gradient Descent (GD) minimizes:

$$L_{train}(\omega) = \frac{1}{N}\sum_{n=1}^{N} e(F(\mathbf{x_n}), y_n)$$

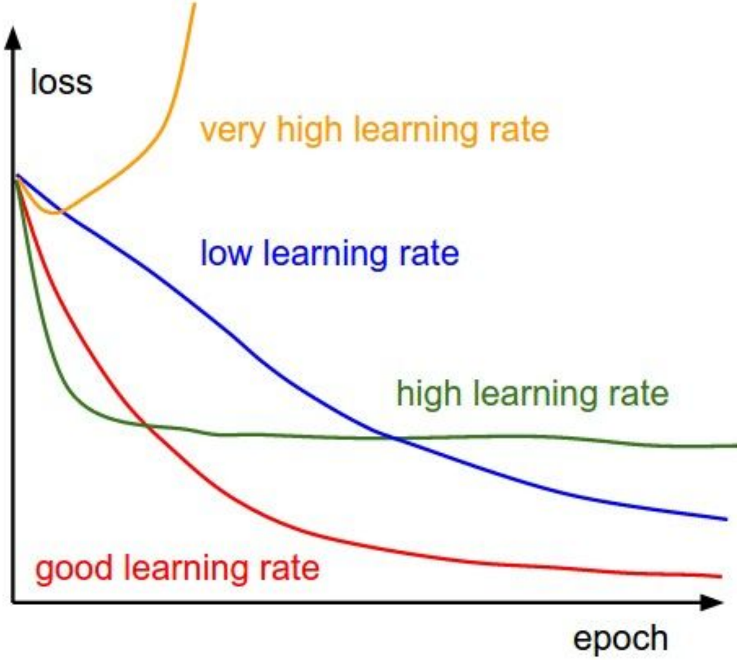
by iterative steps along $-\nabla L_{train}$:

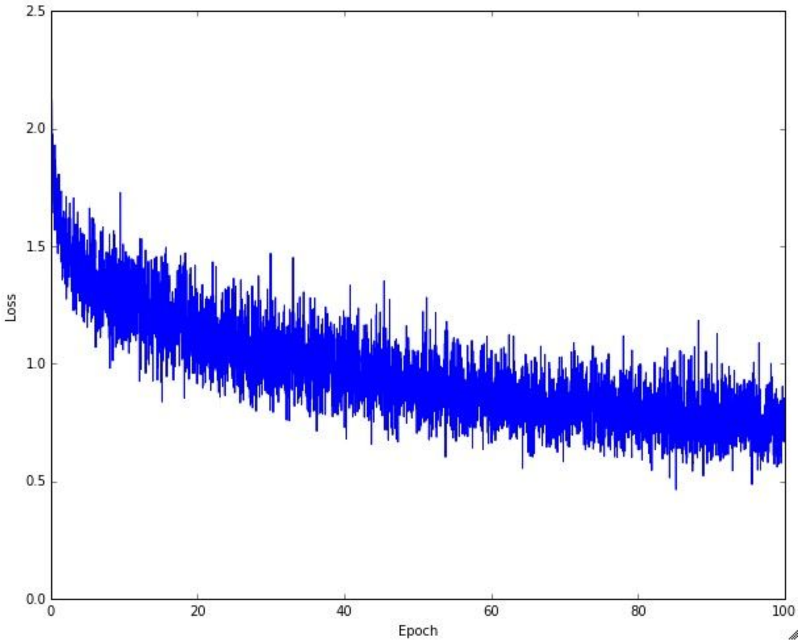
$$\Delta\omega = -\eta\nabla L_{train}(\omega)$$
$$\omega_{prev} = \omega_{next} + \Delta\omega$$

If $\nabla L_{train}(\omega)$ is based on all examples $\{\mathbf{x_n}, y_n\}$ then it is called **batch gradient descent**

images from http://cs231n.stanford.edu/

ML review (NN)



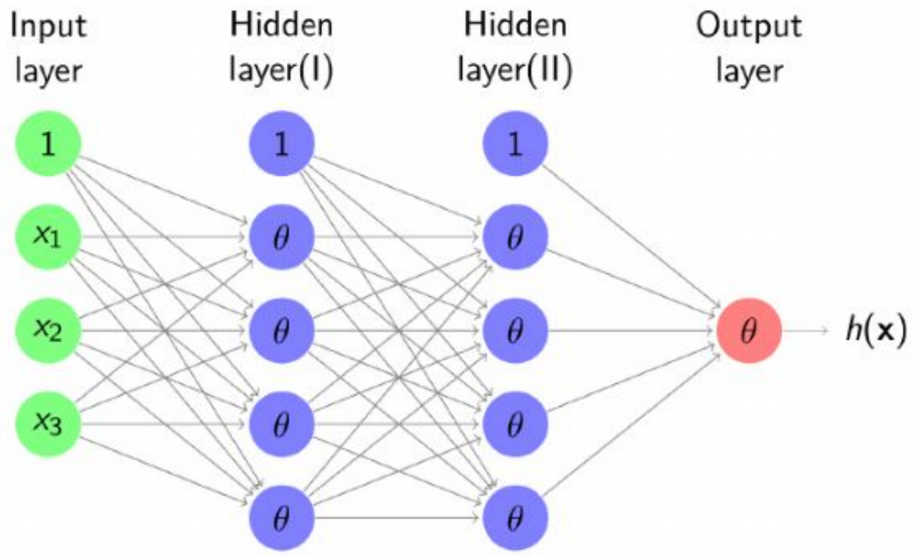


images from http://cs231n.stanford.edu/

Content

- Intro
- ML review
- **Intro to CNN**

Intro to CNN



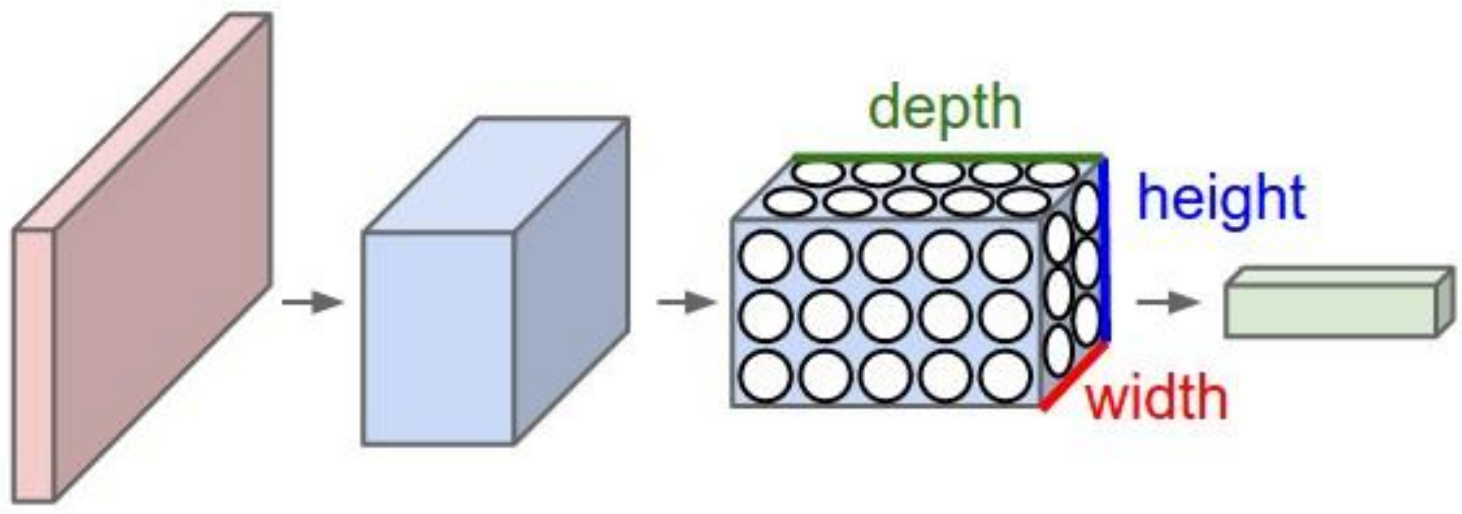
Input layer
$3 \times 256 \times 256 = 200K$

Weights (ω_0):
$200K \times$(size of next layer)

=> millions of parameters
(for just one layer)

Need for alternative architecture

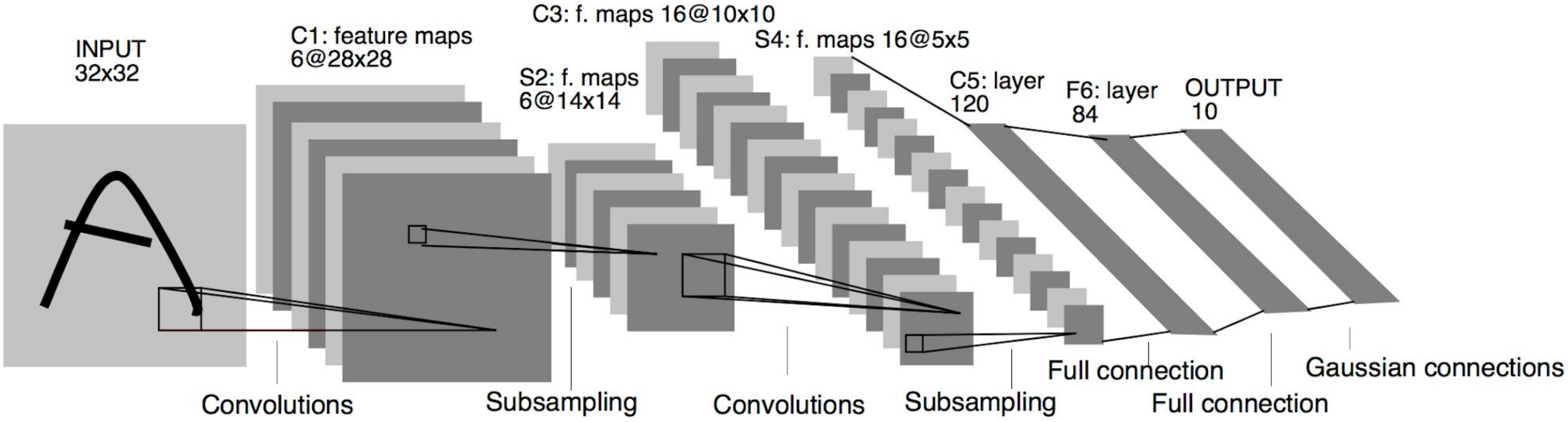
Intro to CNN



images from http://cs231n.stanford.edu/

Intro to CNN

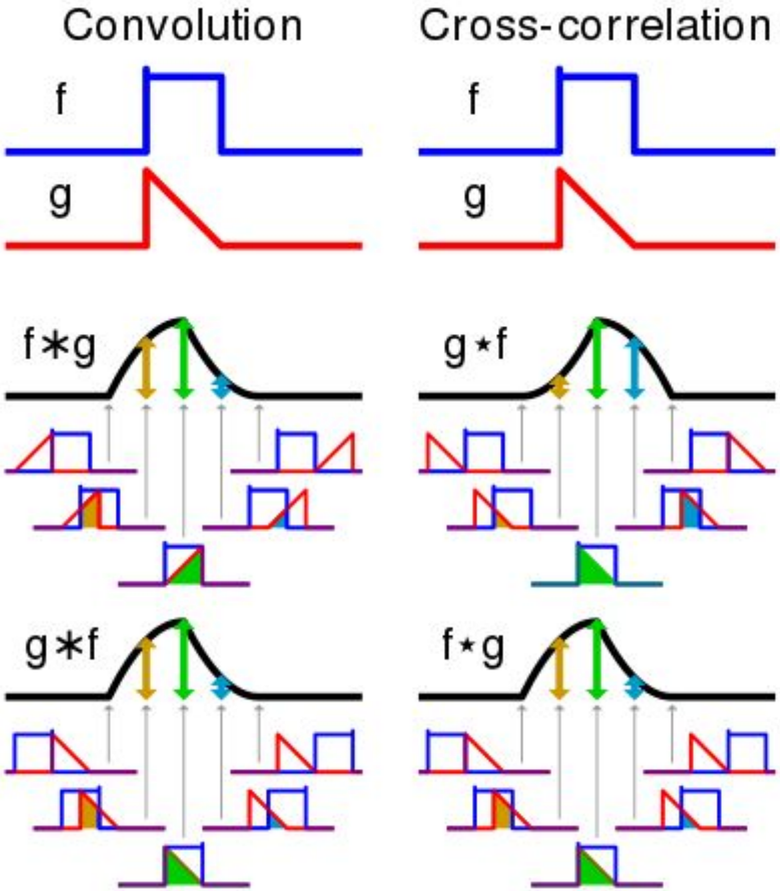
LeNet-5 [1998, paper by LeCun et al.]



Intro to CNN

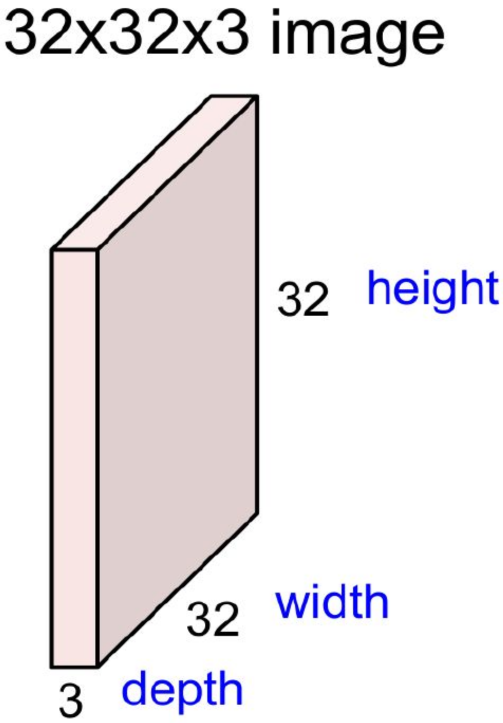
- INPUT holds the raw pixel values of the image.
- $\boxed{\text{CONV layer}}$ computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume.
- POOL layer performs a downsampling operation along the spatial dimensions (width, height).
- FC (i.e. fully-connected) layer computes the class scores. As with ordinary Neural Networks and as the name implies, each neuron in this layer is connected to all the numbers in the previous volume.

Intro to CNN



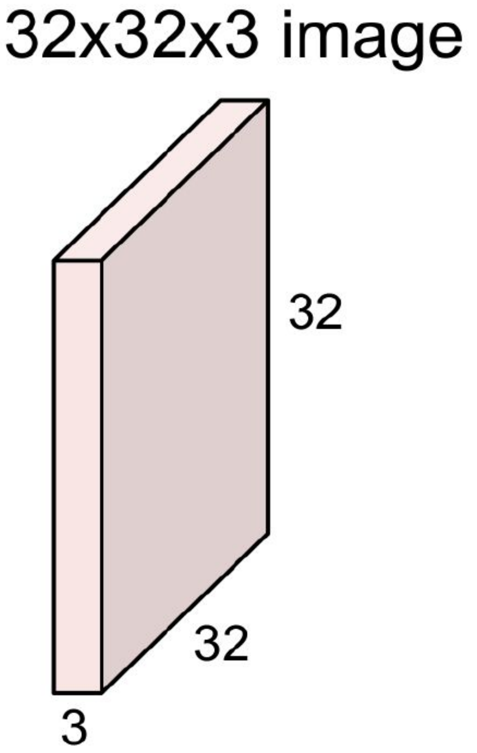
https://en.wikipedia.org/wiki/Convolution

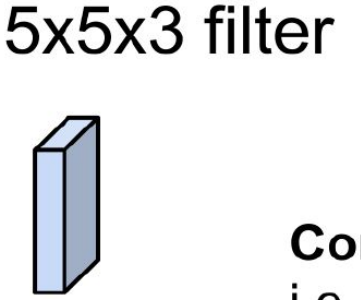
Intro to CNN



images from http://cs231n.stanford.edu/

Intro to CNN



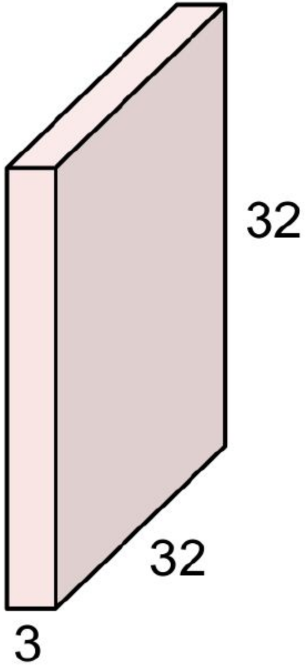


Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

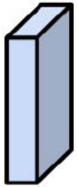
images from http://cs231n.stanford.edu/

Intro to CNN

32x32x3 image



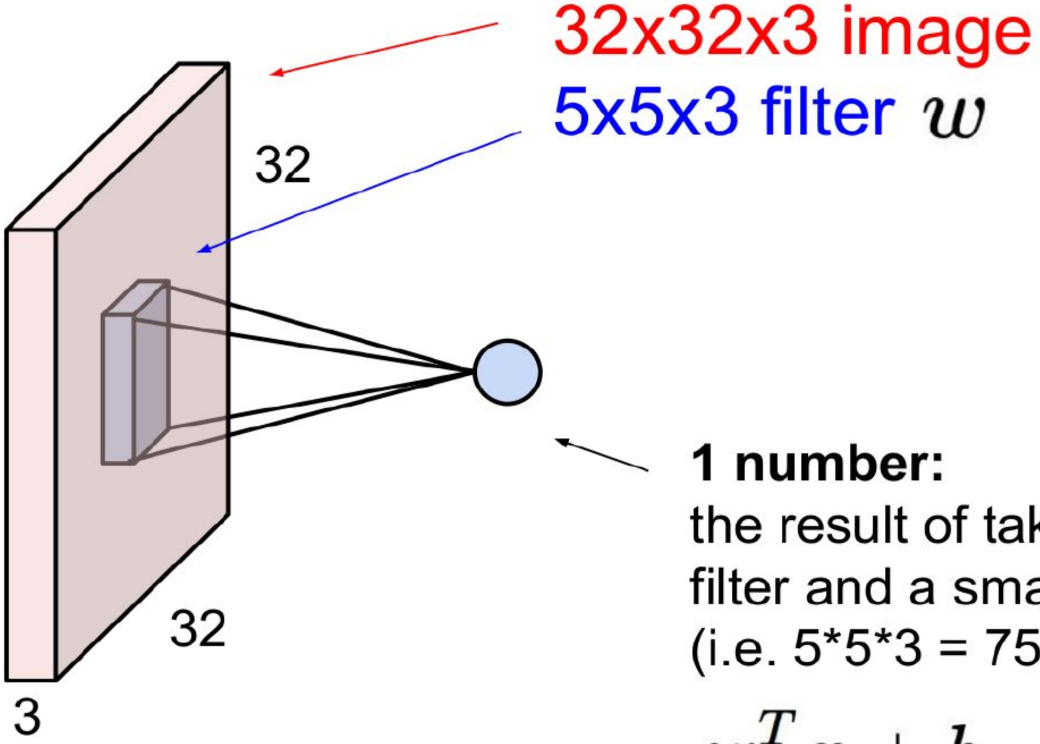
5x5x3 filter



Convolve the filter with the image
i.e. "slide over the image spatially,
computing dot products"

images from http://cs231n.stanford.edu/

Intro to CNN

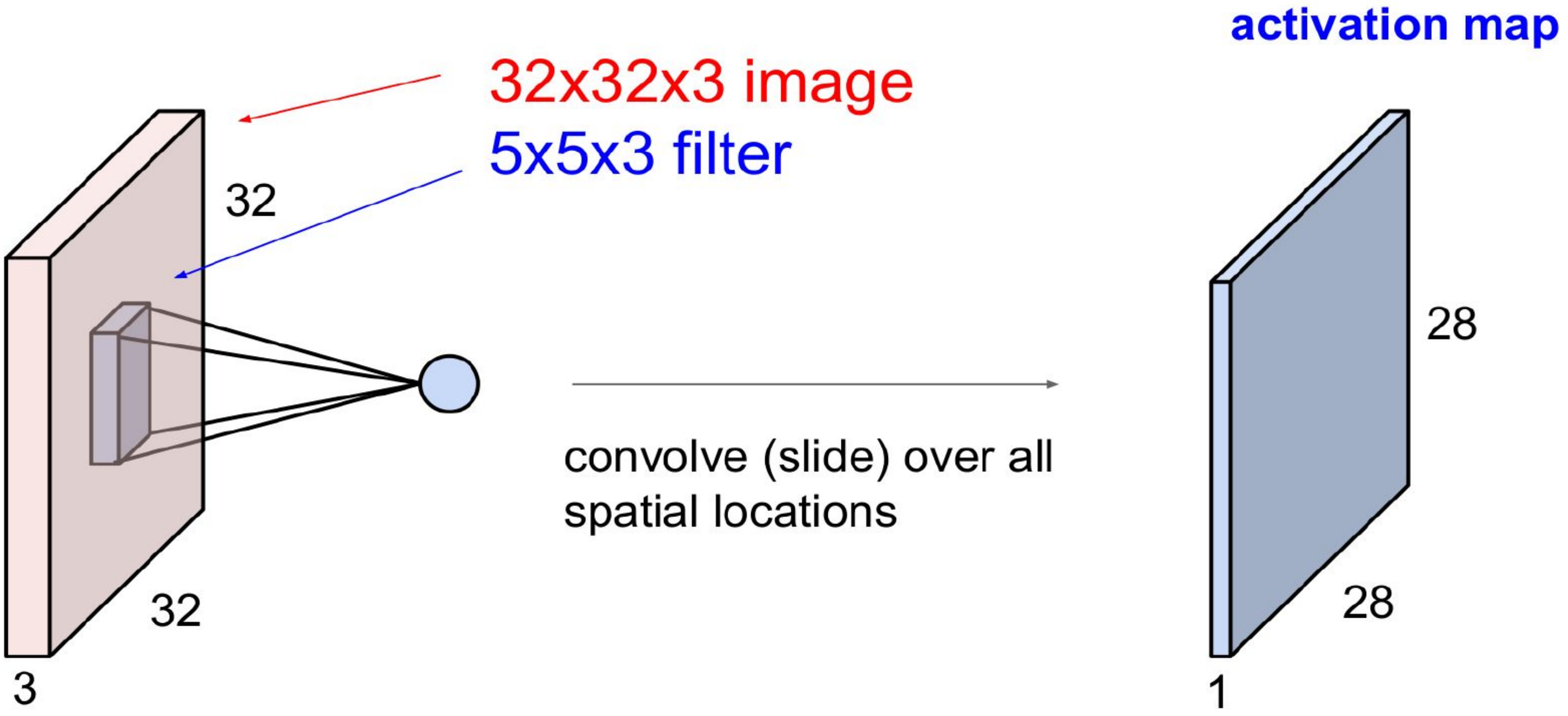


1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

images from http://cs231n.stanford.edu/

Intro to CNN

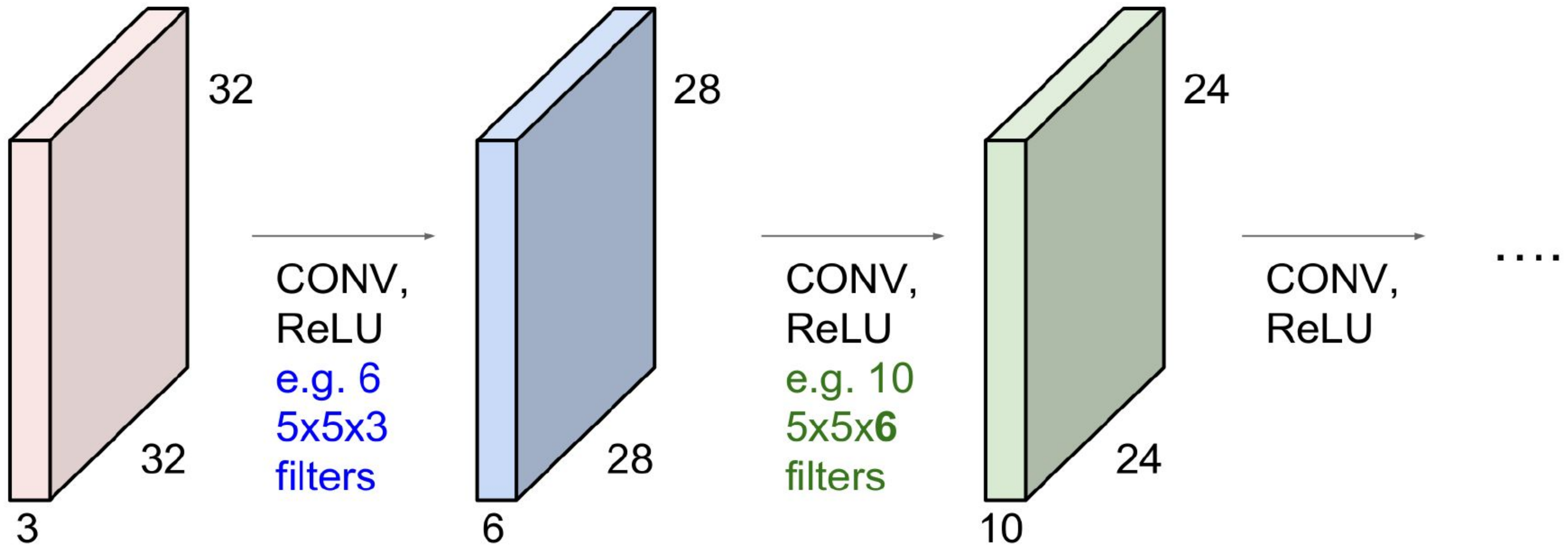


images from http://cs231n.stanford.edu/

Intro to CNN

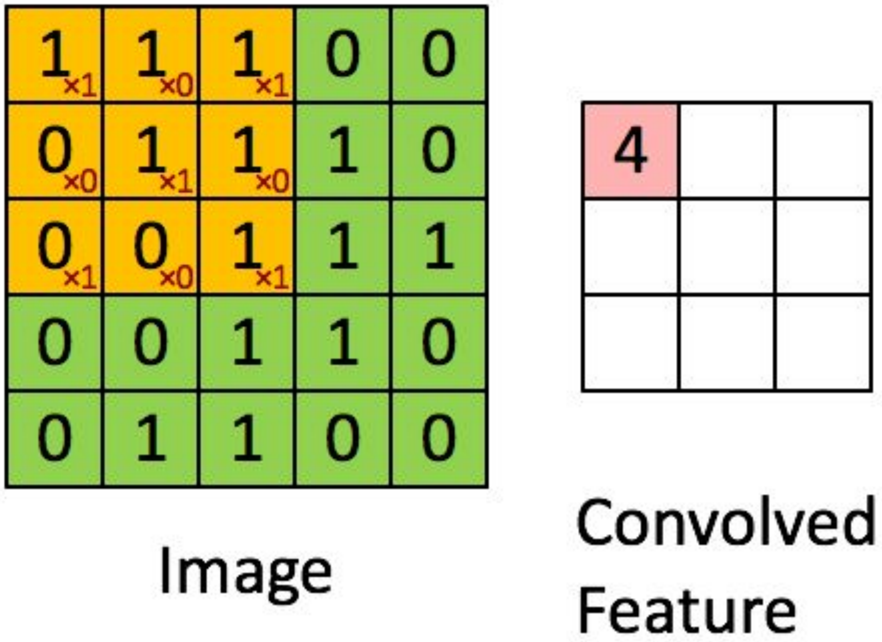
images from http://cs231n.stanford.edu/

Intro to CNN



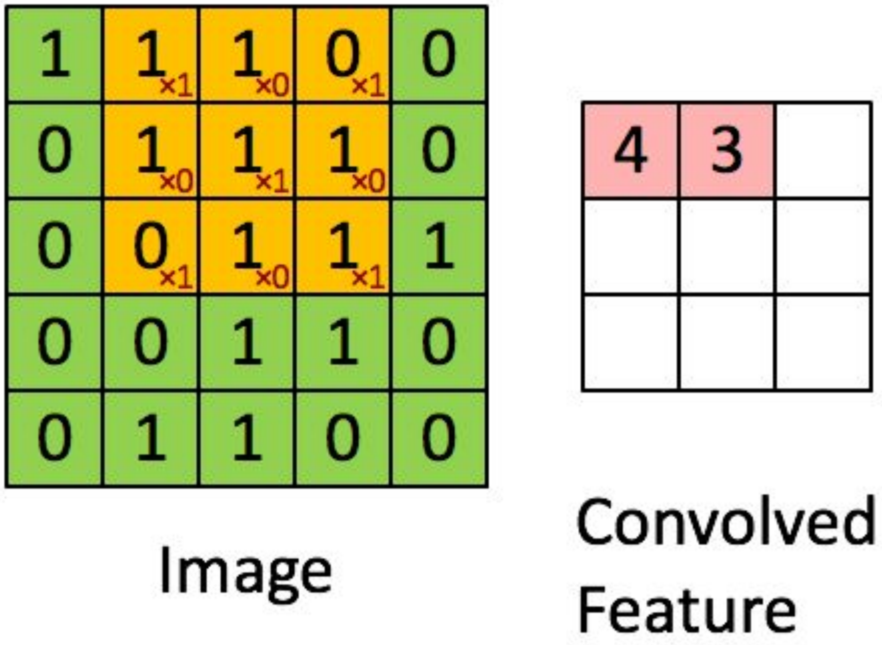
images from http://cs231n.stanford.edu/

Intro to CNN



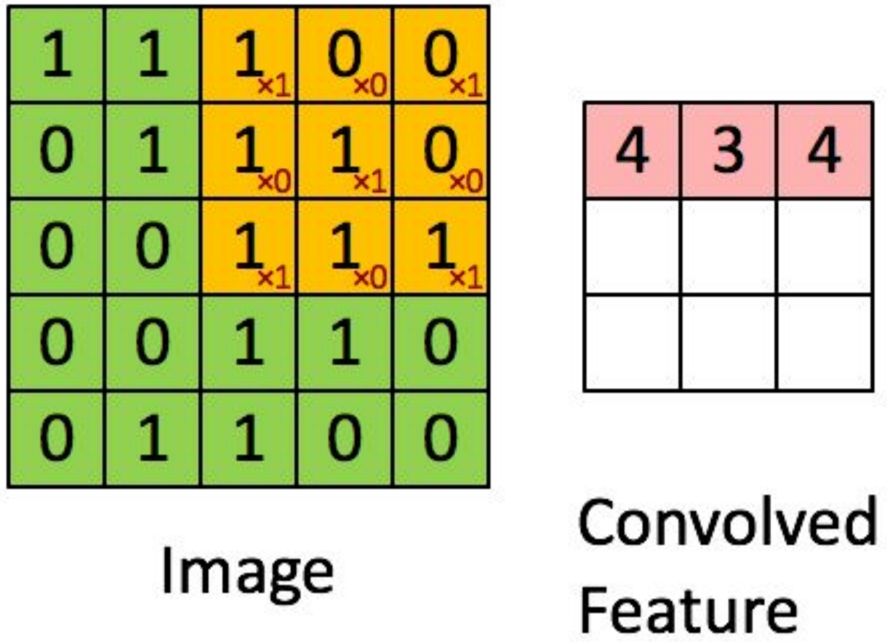
images from http://cs231n.stanford.edu/

Intro to CNN



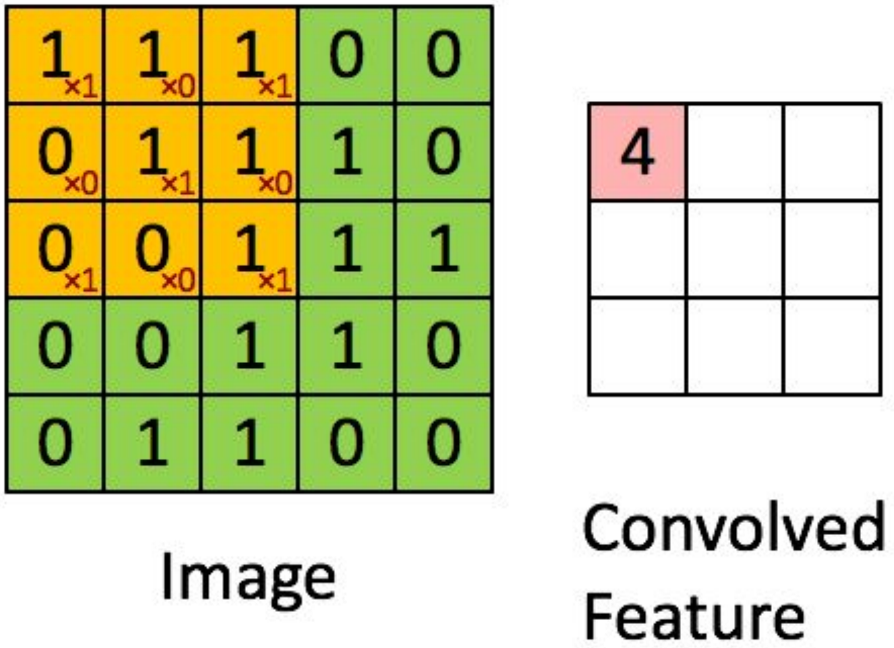
images from http://cs231n.stanford.edu/

Intro to CNN



images from http://cs231n.stanford.edu/

Intro to CNN


1 ×1
1 ×0
1 ×1
0
0
0 ×0
1 ×1
1 ×0
1
0
0 ×1
0 ×0
1 ×1
1
1
0
0
1
1
0
0
1
1
0
0
Image
4
Convolved
Feature

Intro to CNN

We can use one single convolutional layer to modify a certain image

[1. 1. 1.]
[1. 1. 1.]
[1. 1. 1.]

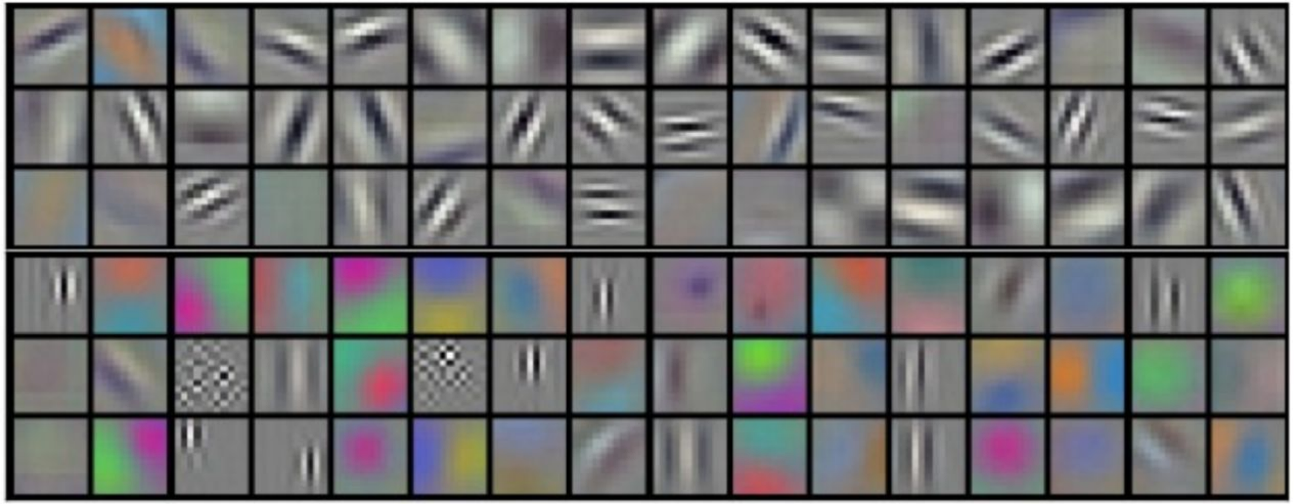
[1. 2. 1.]
[0. 0. 0.]
[-1. -2. -1.]

[0. -1. 0.]
[-1. 5. -1.]
[0. -1. 0.]

based on CS 20SI: Tensorflow for Deep Learning Research

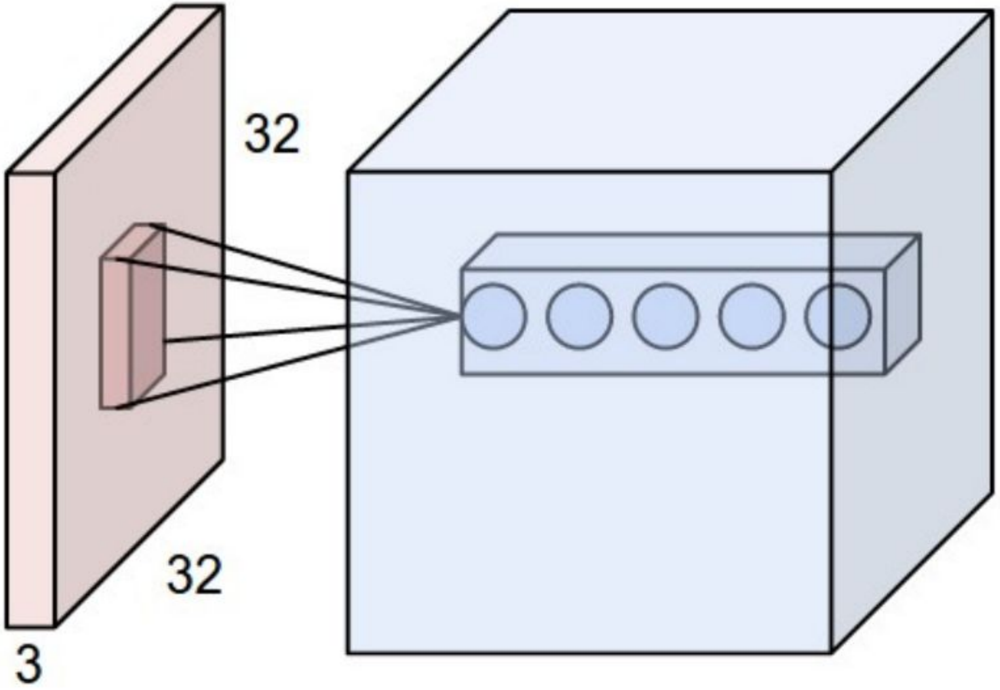
Intro to CNN

In training, we don't specify kernels.
We learn kernels!



images from http://cs231n.stanford.edu/

Intro to CNN



images from http://cs231n.stanford.edu/

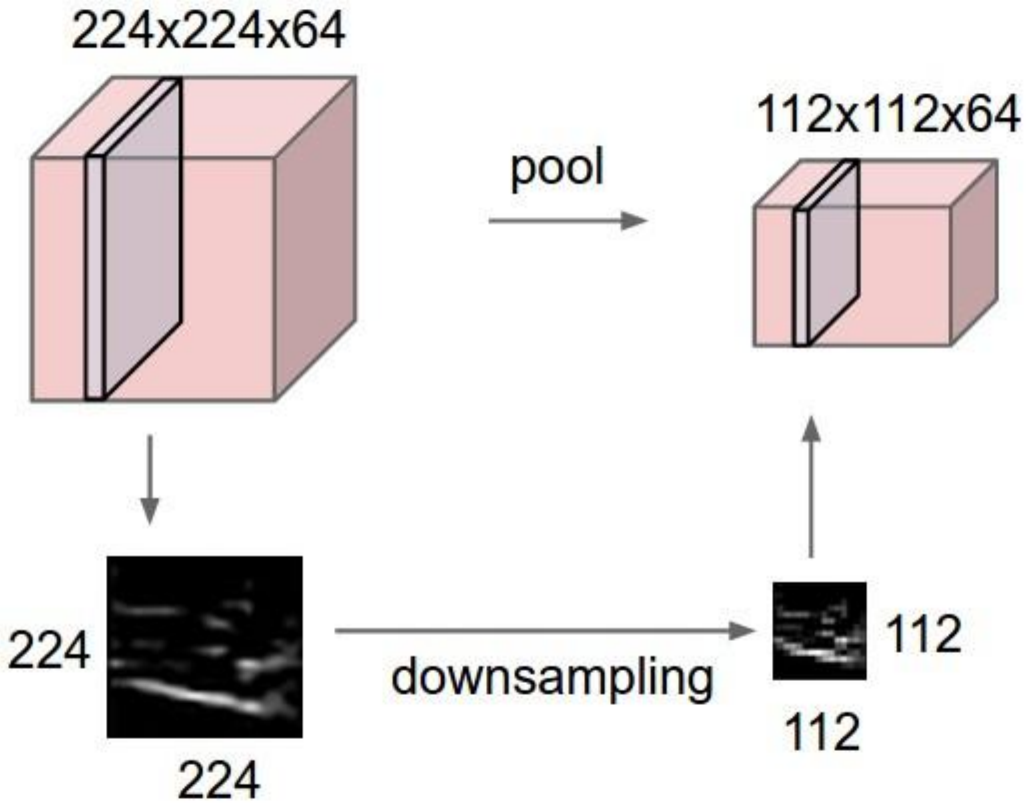
Intro to CNN

- Accepts a volume of size $W1 \times H1 \times D1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W2 \times H2 \times D2$ where:
  - $W2 = (W1 - F + 2P)/S + 1$,
  - $H2 = (H1 - F + 2P)/S + 1$
  - $D2 = K$
- With parameter sharing, it introduces $F \times F \times D1$ weights per filter, for a total of $(F \times F \times D1) \times K$ weights and K biases.

Intro to CNN

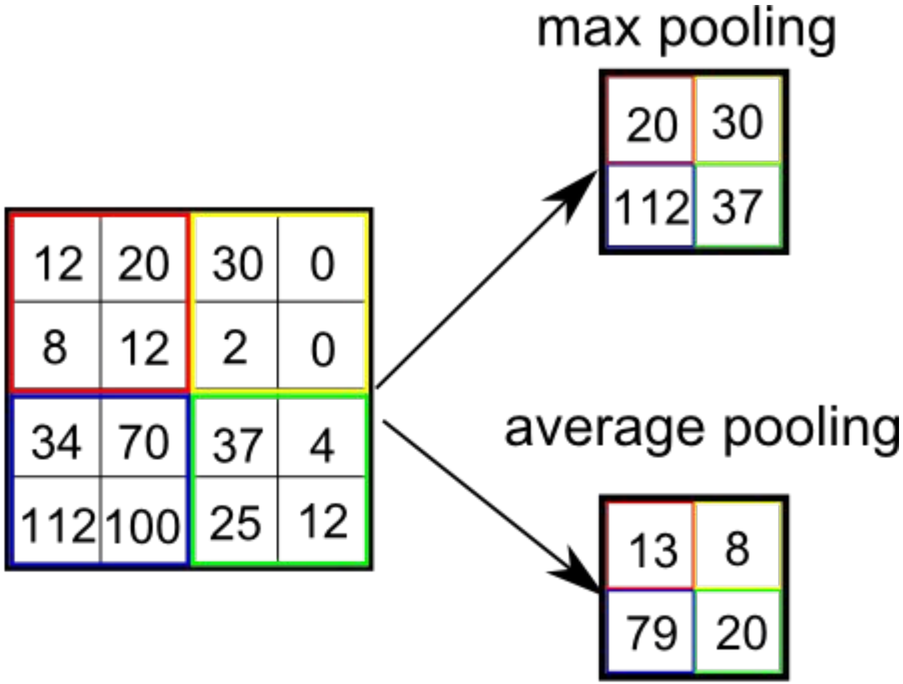
- INPUT holds the raw pixel values of the image.
- CONV layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume.
- POOL layer performs a downsampling operation along the spatial dimensions (width, height).
- FC (i.e. fully-connected) layer computes the class scores. As with ordinary Neural Networks and as the name implies, each neuron in this layer is connected to all the numbers in the previous volume.

Intro to CNN



images from http://cs231n.stanford.edu/

Intro to CNN



images from http://cs231n.stanford.edu/

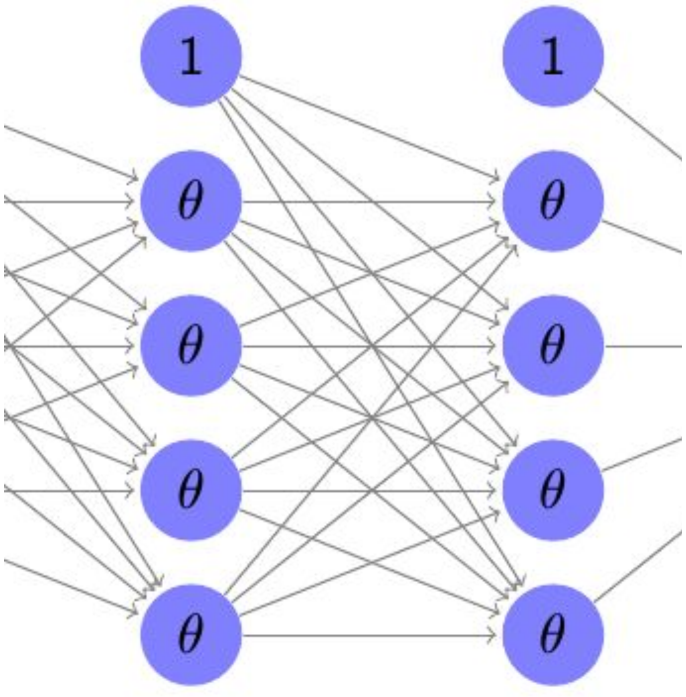
Intro to CNN

- Accepts a volume of size $W1 \times H1 \times D1$
- Requires three hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W2 \times H2 \times D2$ where:
  - $W2 = (W1 - F)/S + 1$
  - $H2 = (H1 - F)/S + 1$
  - $D2 = D1$

Intro to CNN

- INPUT holds the raw pixel values of the image.
- CONV layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume.
- POOL layer performs a downsampling operation along the spatial dimensions (width, height).
- $\boxed{\text{FC (i.e. fully-connected) layer}}$ computes the class scores. As with ordinary Neural Networks and as the name implies, each neuron in this layer is connected to all the numbers in the previous volume.

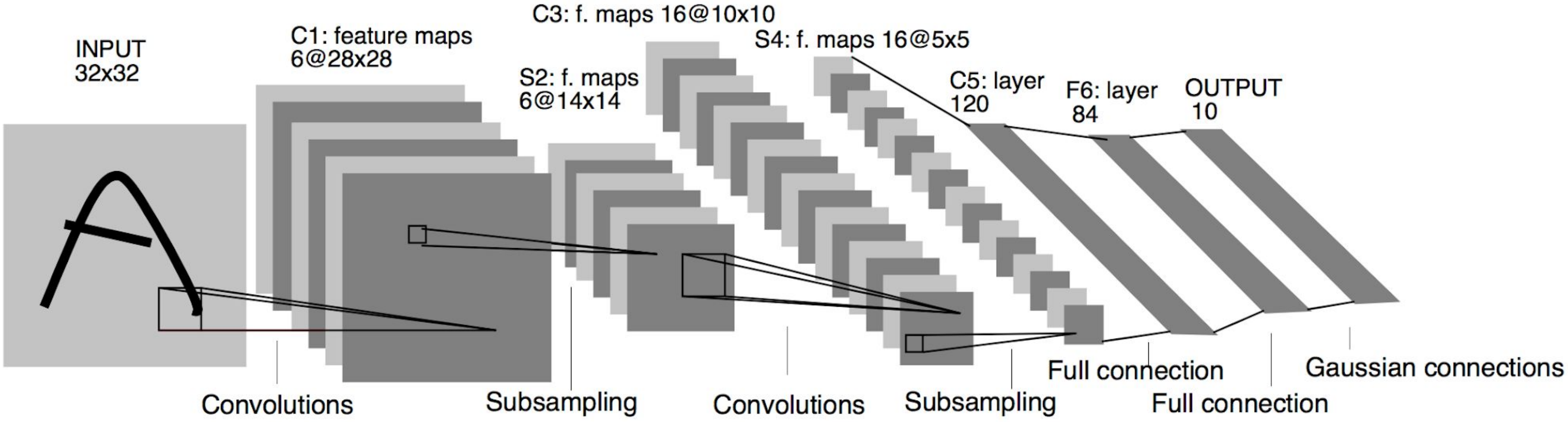
Intro to CNN



1
1
θ
θ
θ
θ
θ
θ
θ
θ
θ
θ

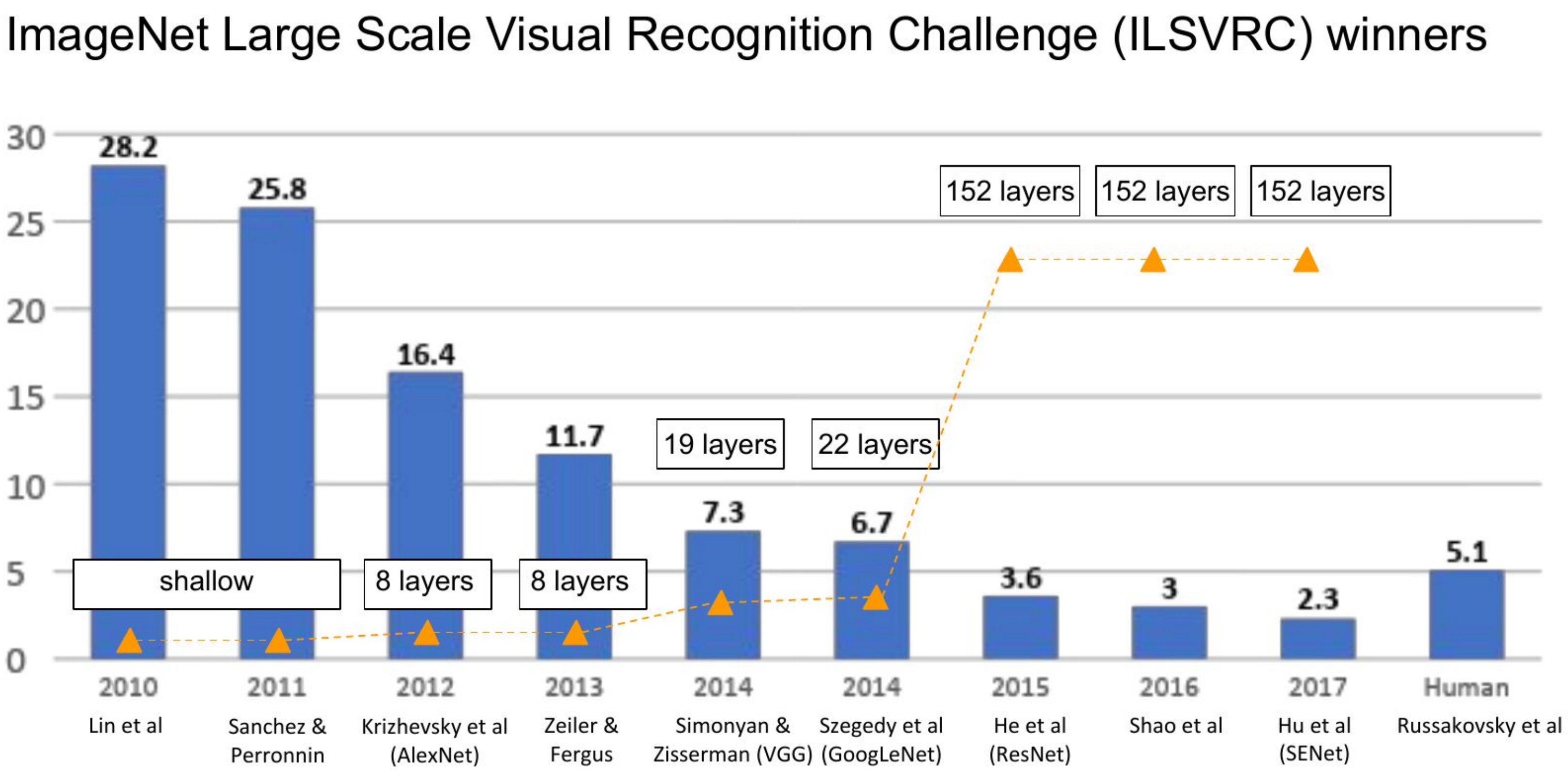
Intro to CNN

LeNet-5 [1998, paper by LeCun et al.]



Intro to CNN

Intro to CNN



images from http://cs231n.stanford.edu/

Summary

- INPUT holds the raw pixel values of the image.
- CONV layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume.
- POOL layer performs a downsampling operation along the spatial dimensions (width, height).
- FC (i.e. fully-connected) layer computes the class scores. As with ordinary Neural Networks and as the name implies, each neuron in this layer is connected to all the numbers in the previous volume.