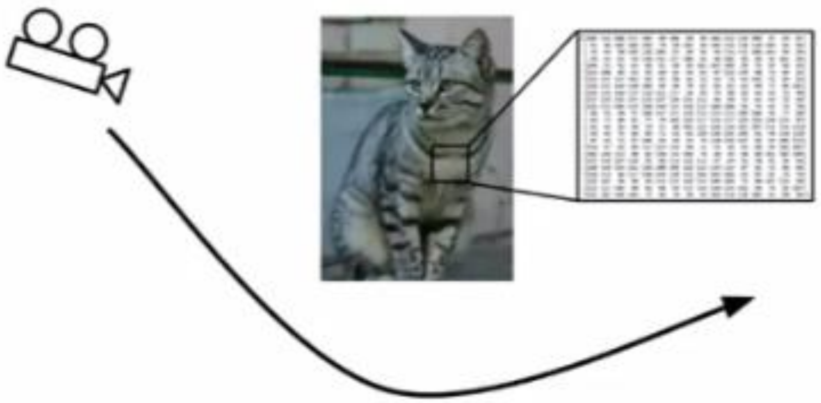
ML-101: Neural Networks

BY SARTHAK CONSUL

Viewpoint



Illumination

Deformation

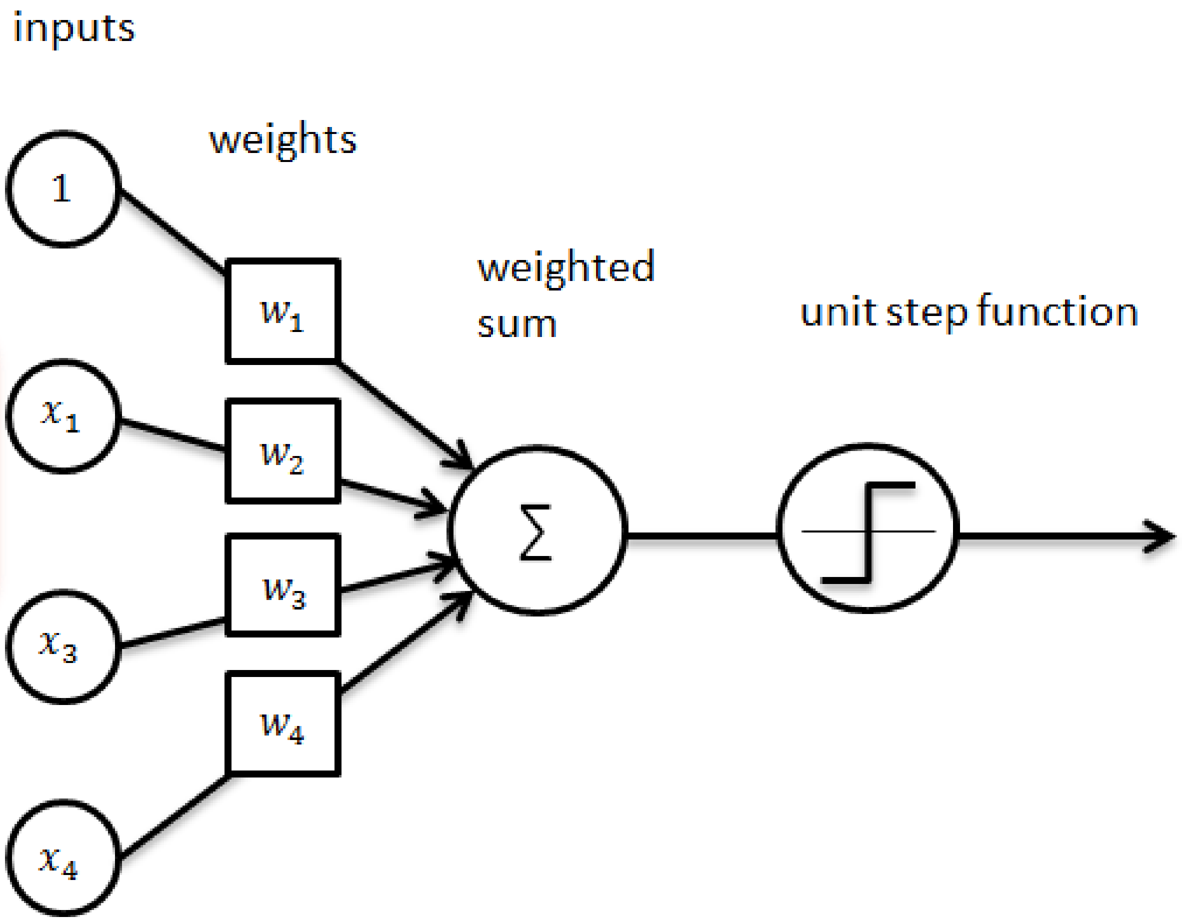
Occlusion

Clutter

Intraclass Variation

Perceptron – predecessor to the neuron

Focus on ideal weight not best output



Frank Rosenblatt, 1957

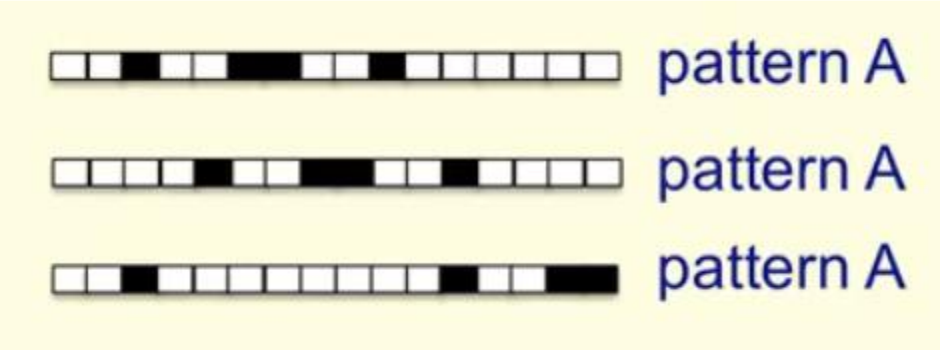
Problems with the Perceptron

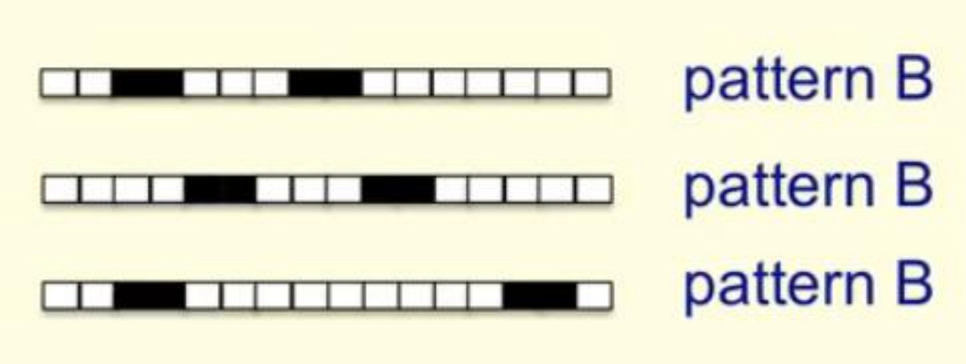
- Cannot learn unless right features are used.
  - Need to choose by hand enough features - IMPRACTICAL
- Eg. XNOR: Positive Cases (1,1), (0,0) ; Negative Cases (1,0), (0,1)

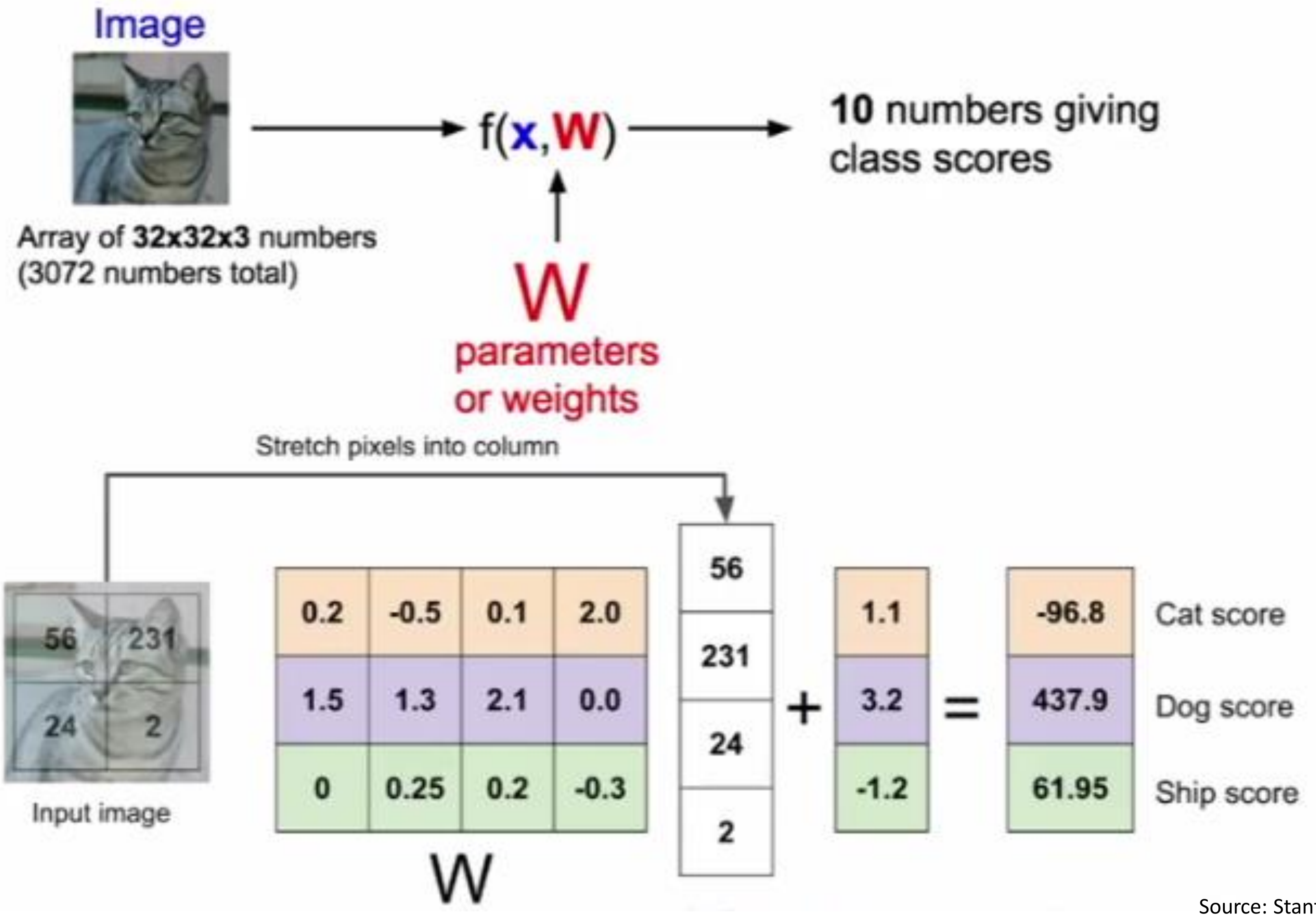
Can't

SIGMOID

- Pattern differentiation (of same pixels and with wrap around)

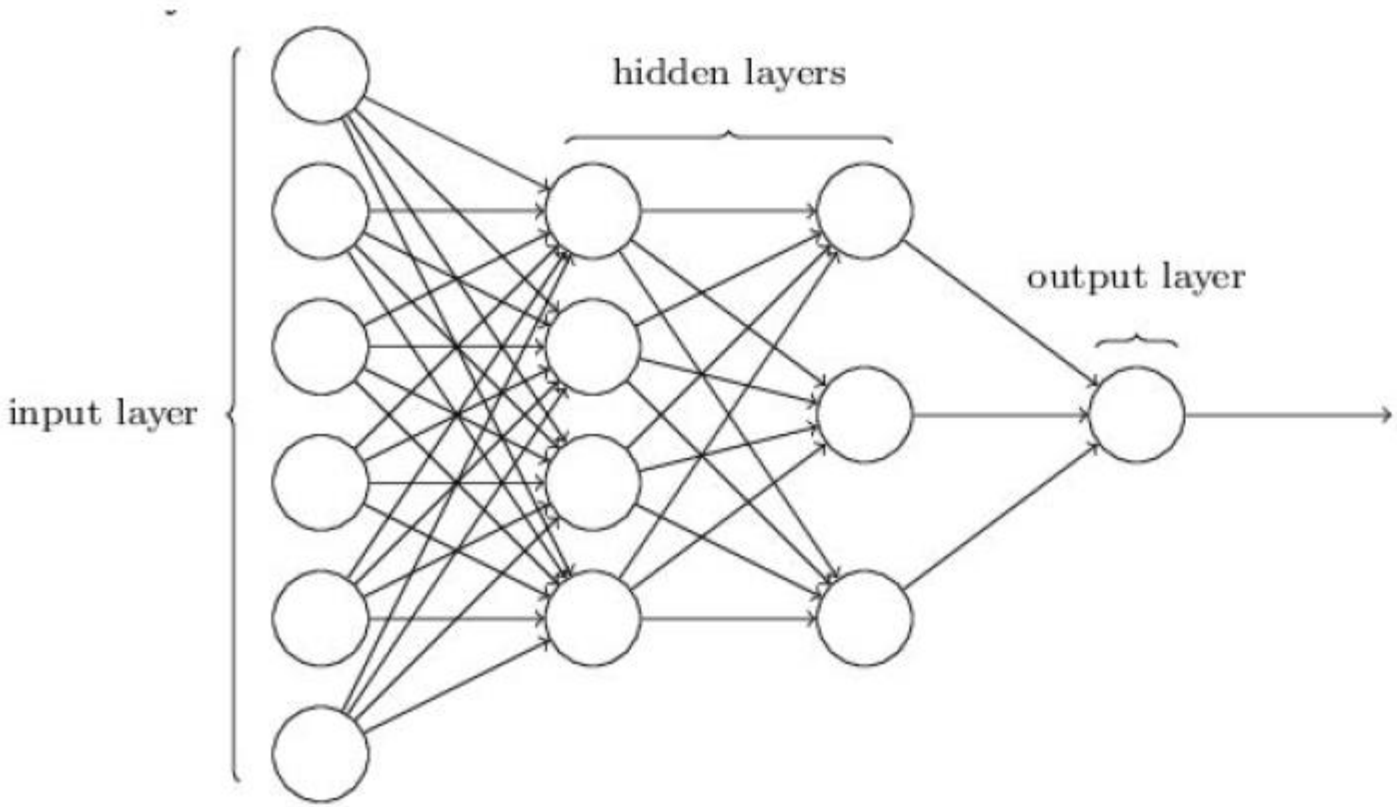






Source: Stanford CS231n – CNN for CV

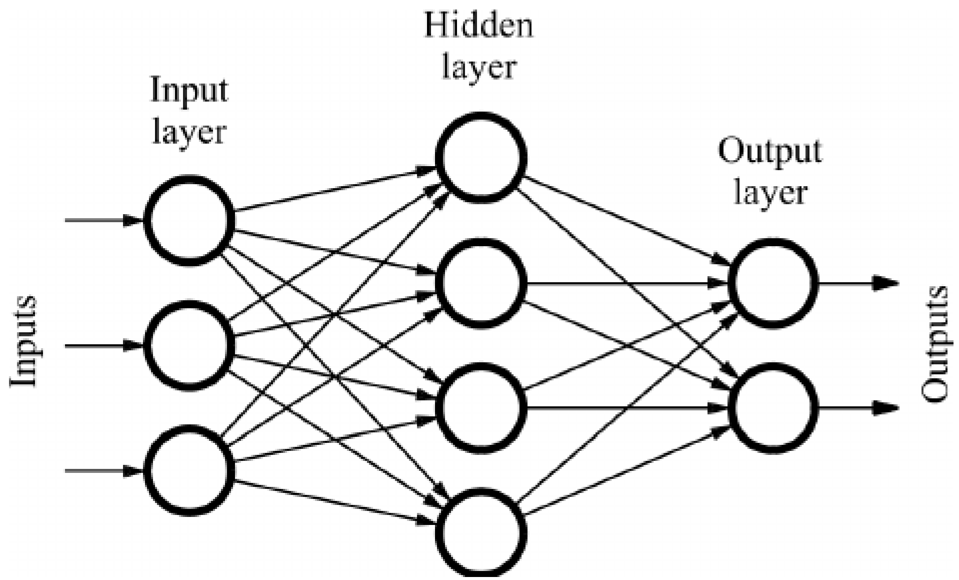
Components of the Neural Network

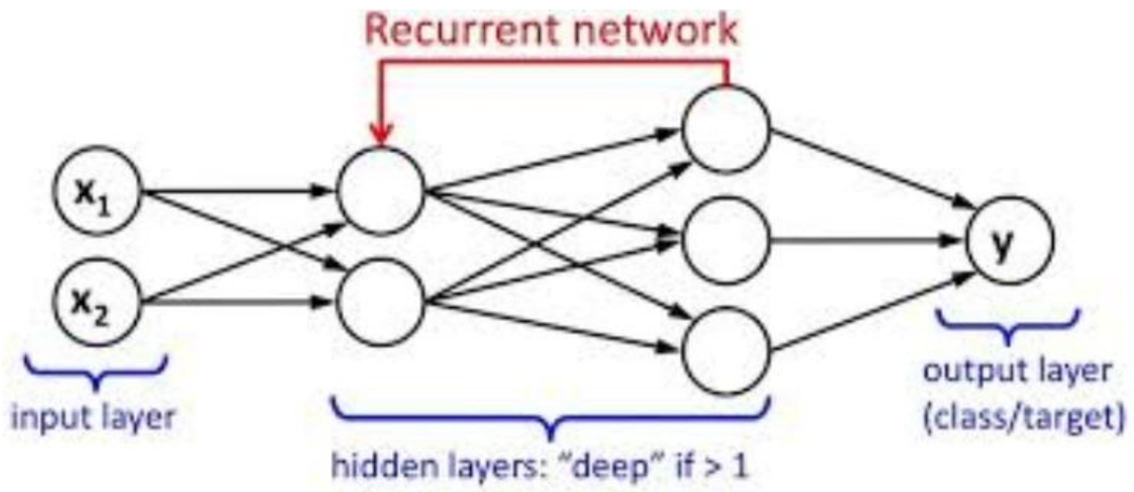


- Biases & Weights
- Activation Function
- Layers
- Neurons

Neural Network Architectures

- Feedforward Network
- Recurrent Network





Softmax Classifier (Multinomial logistic Regression)

- A 'soft' max function (s=log probabilities of class)

$$P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

- Want to maximize the log likelihood
- Loss Function

$$L_i = -\log P(Y = y_i|X = x_i)$$

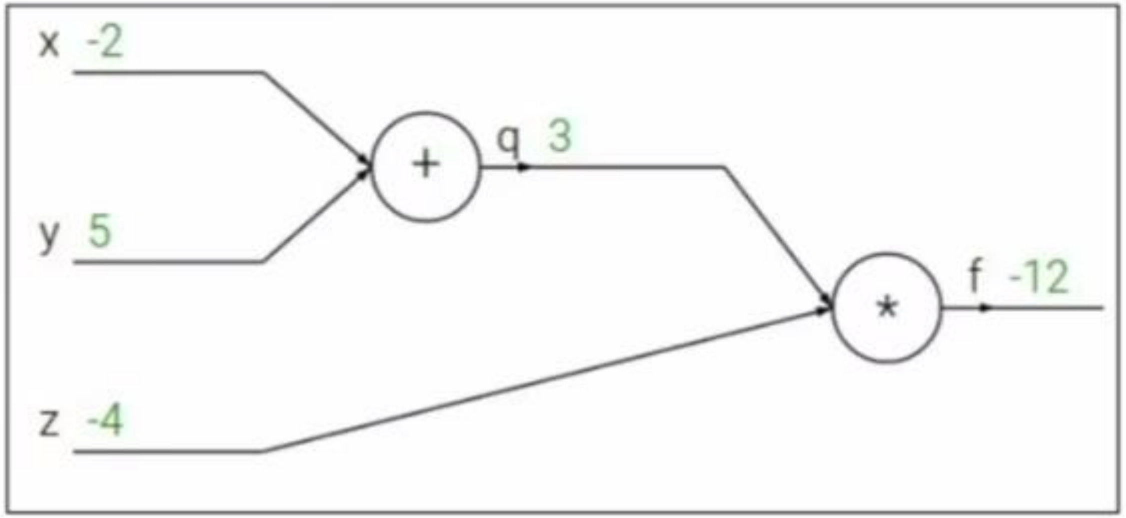
$$\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$$

Backpropagation

- Essentially Chain Rule
- Eg. $f(x, y, z) = (x + y).z$

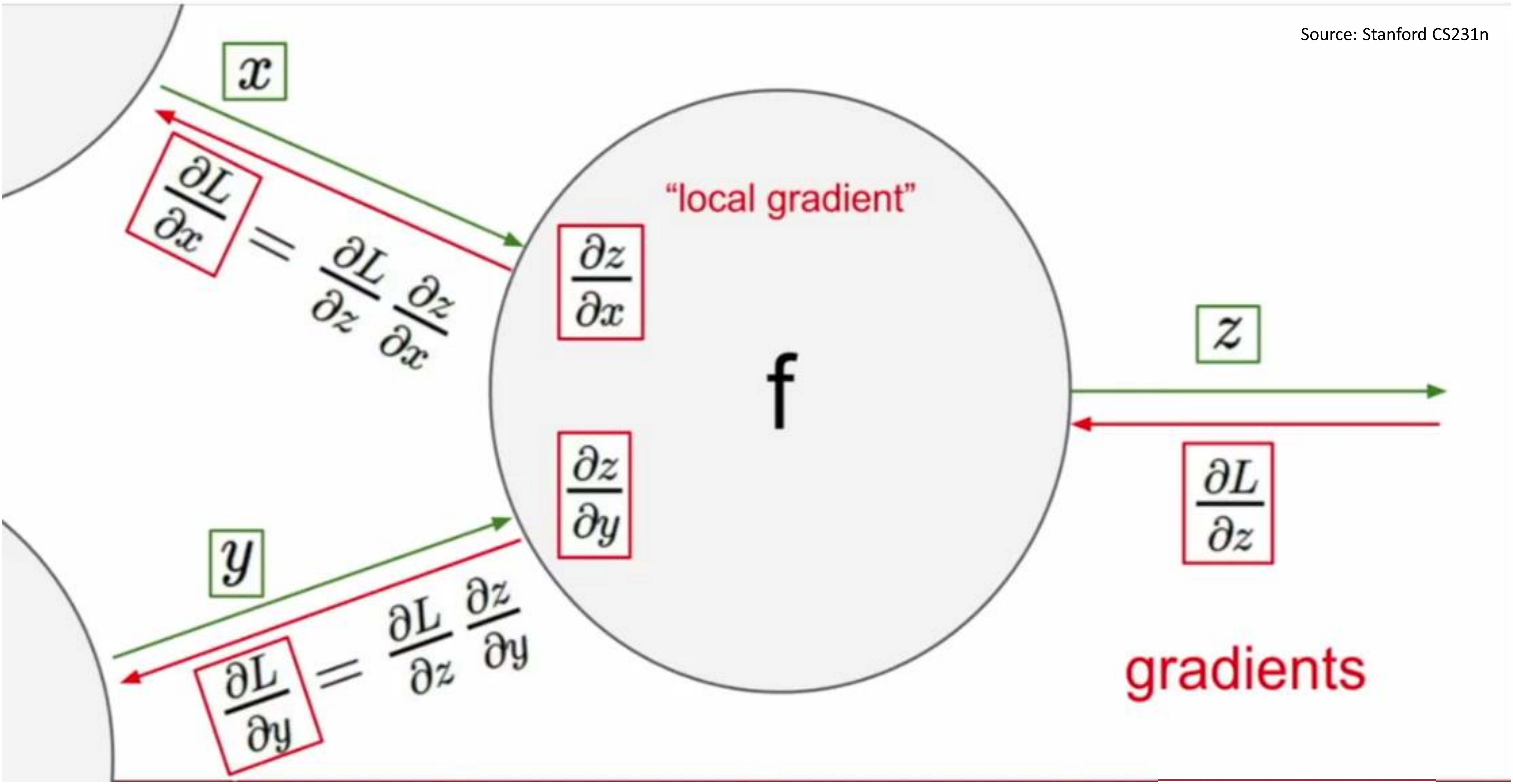
$$q(x, y) = x + y \quad f = q.z$$

$$\frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1 \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$



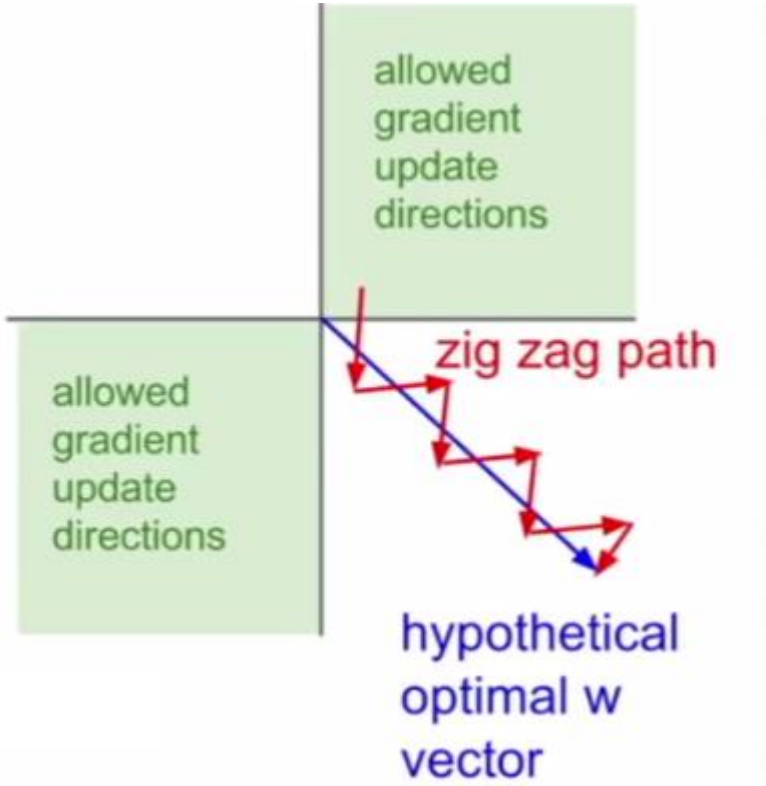
- For another step by step example: https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/

Source: Stanford CS231n

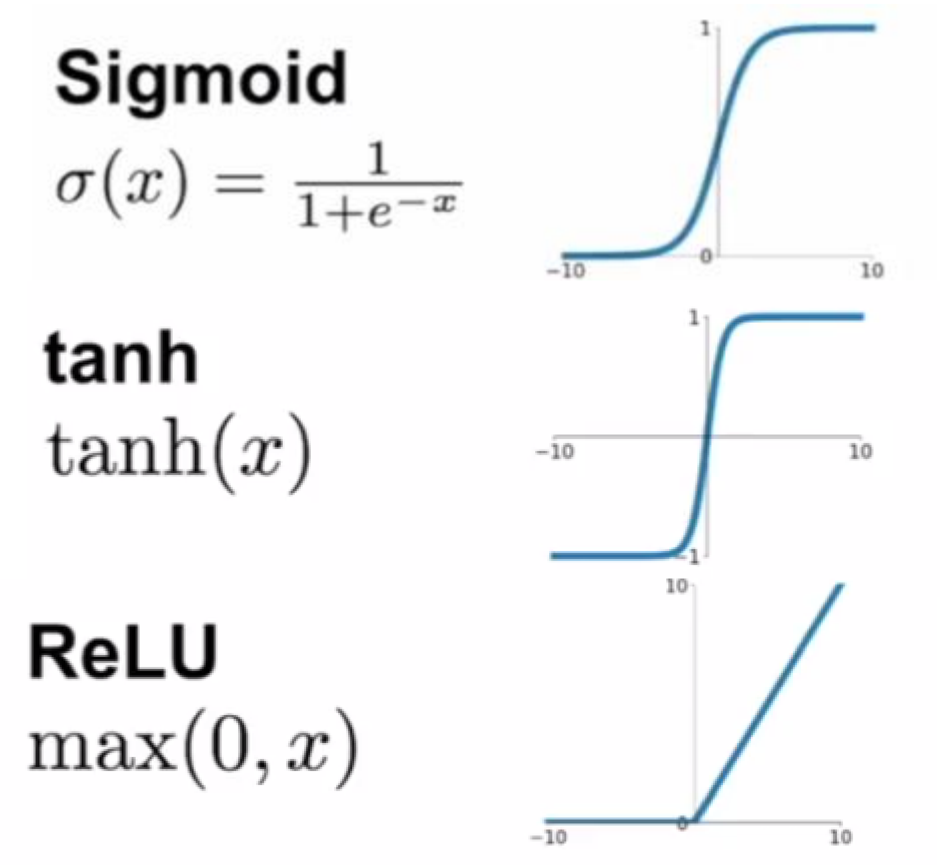


Why Zero Mean Data?

- Suppose input is always positive
- $f(w^T x + b)$
- $\frac{\partial f}{\partial w} = x \geq 0$
- Gradient $\frac{dy}{dw}$, always positive or negative
- Inefficient Gradient Updates!



Activation functions



o squashes range to (0,1)
+ nice interpretation of saturating firing rate of neuron
- Gradients ‘killed’ at saturation
- not zero centered
- exp() is computationally expensive

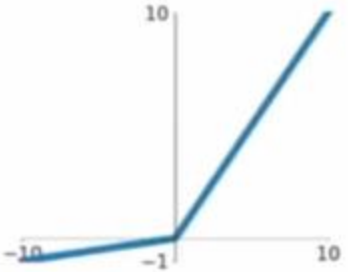
o squashes range to (-1,1)
+ zero centered
- Gradients ‘killed’ at saturation

+ Does not saturate (for x>0)
+ Computationally efficient
+ Faster convergence (6x)
+ Closer approximation to biological neurons
-not zero centered
- Gradient saturation in –ve region

Activation functions contd.

Leaky ReLU

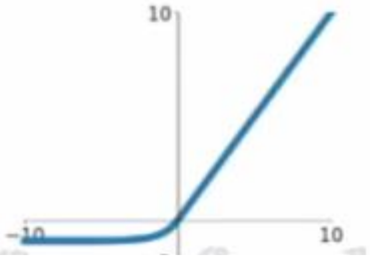
$\max(0.1x, x)$



+ neuron will not 'die'
o PReLU has parameter of slope for x<0
+closer to zero centering

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



+ closer to zero centering
- computation requires exp()
+ robustness to noise
+ does not saturate

Maxout

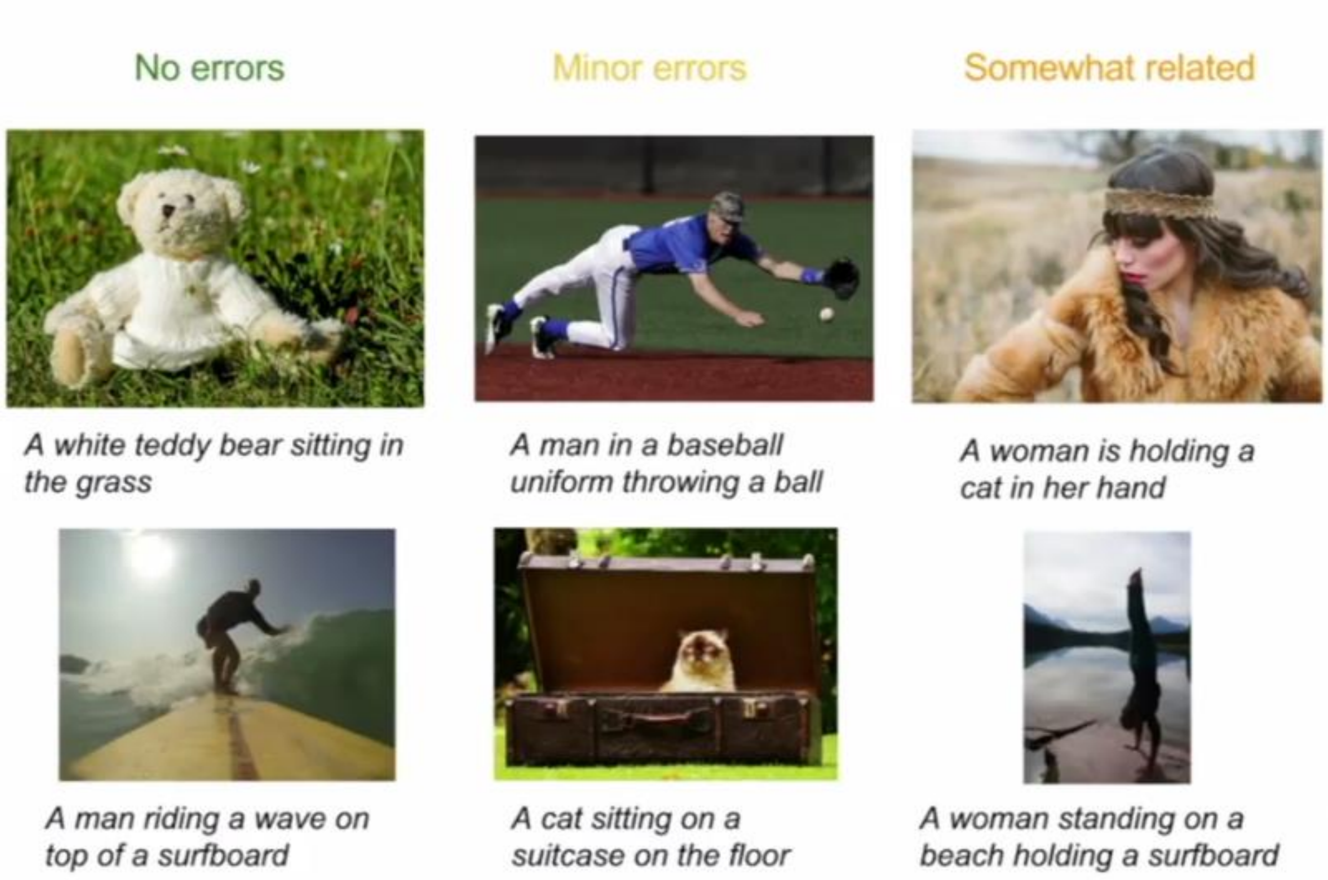
$\max(w_1^T x + b_1, w_2^T x + b_2)$

+ generalizes ReLU and Leaky ReLU
+ non-linearity
- Twice the parameters

ConvNets

- Image Classification
- Image Retrieval
- Detection
- Segmentation
- Image Captioning

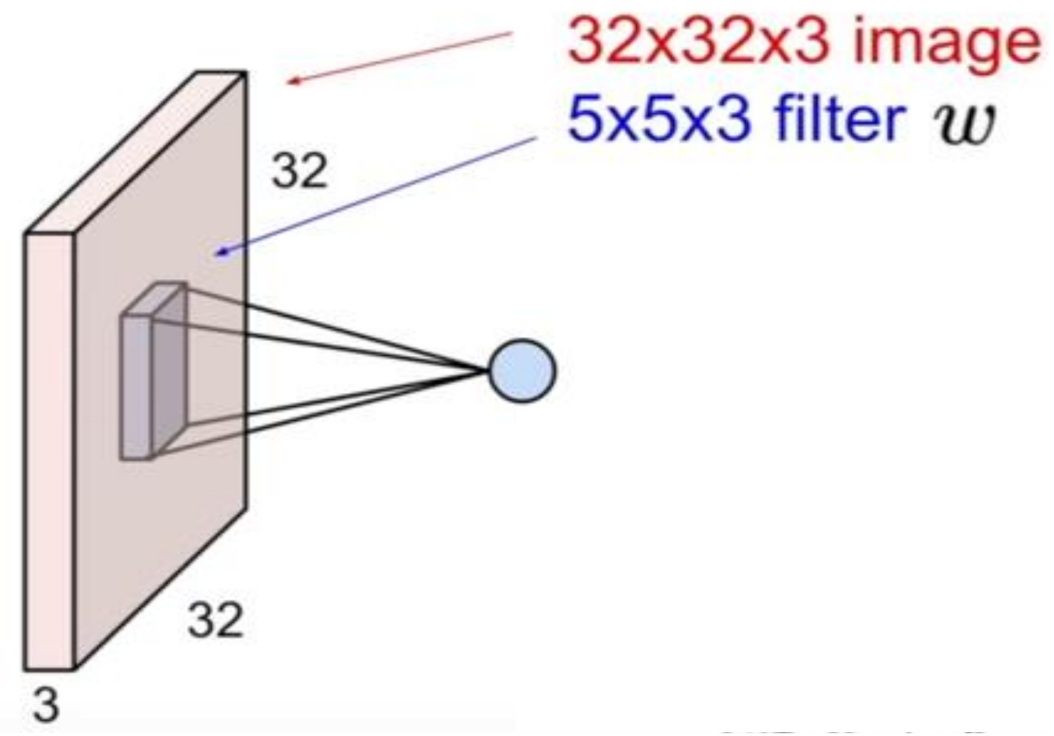
- Fun Stuff!
  - DeepArt
  - Self Driving Cars
  - Street sign recognition



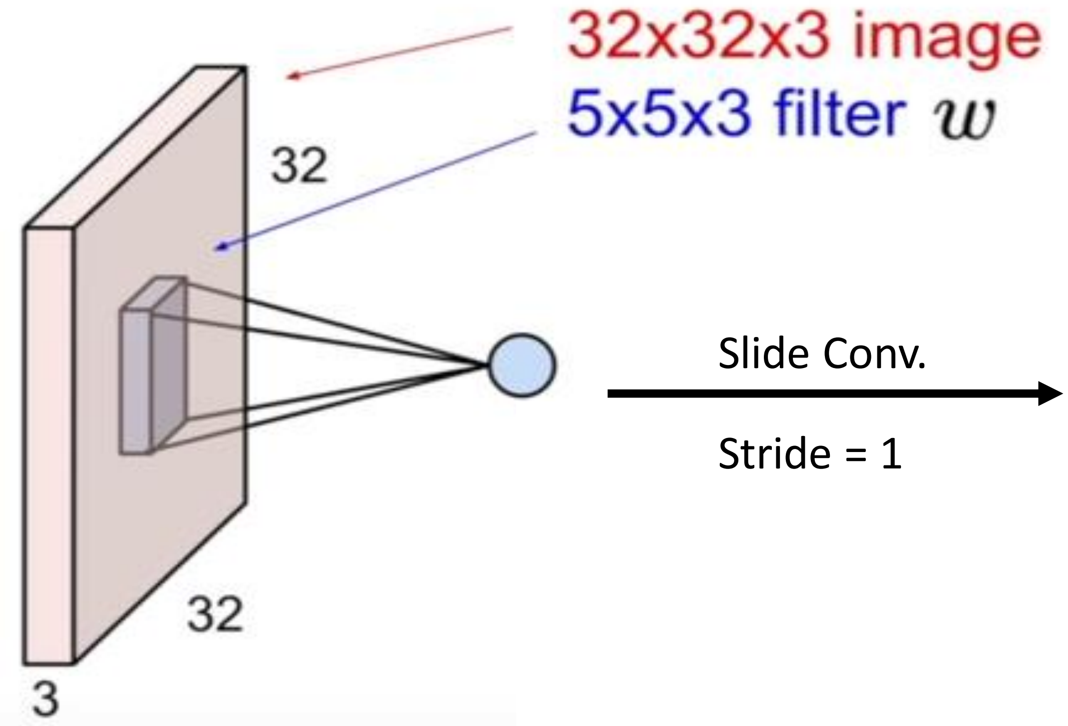
Source: Justin Johnson

Convolution

- Image of size $32 \times 32 \times 3$
- 'Slide' filter size $5 \times 5 \times 3$ (Conv5)
- Similar to Grey Transforms

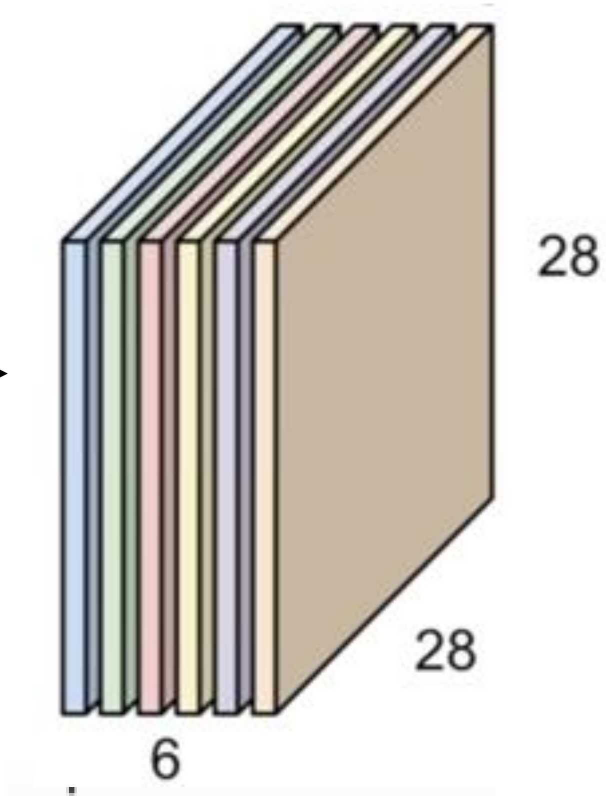


Convolution



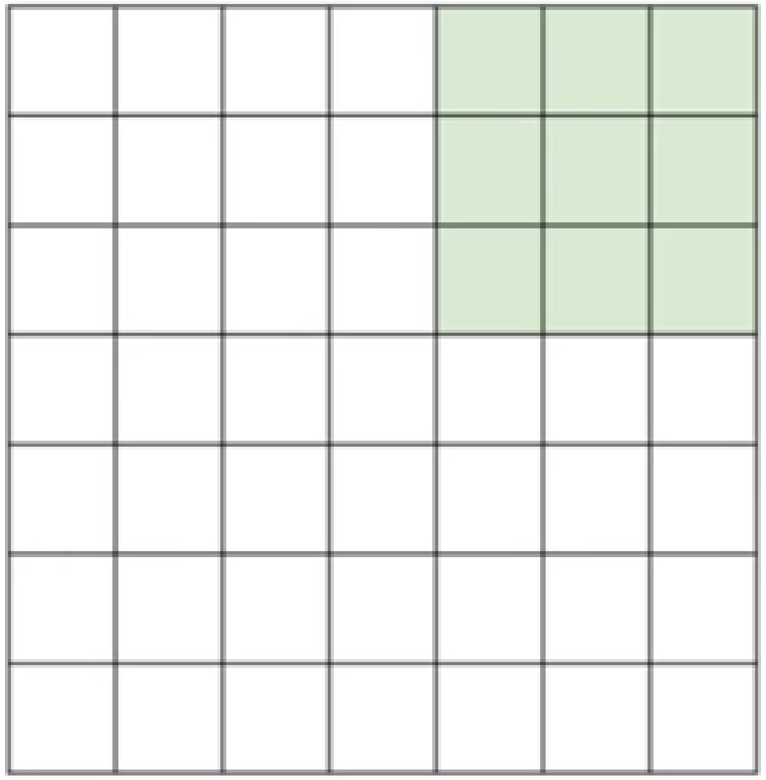
Slide Conv.

Stride = 1



Stride in Conv. Layers

- Input of size 7×7
- 3×3 (Conv3) with stride = 2
- Output Size = **(N-F)/Stride + 1**



Max Pooling

- Makes layer more manageable
  - One method of Downsampling
- Input of size 4×4
- 2×2 MaxPool with stride = 2
- Output Size = **(N-F)/Stride + 1**

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

Convolutional Network

- ConvNet is sequence of convolutional layers, interspersed with activation functions
  - [(Conv-ReLU)*N-POOL]*M – (FC-ReLU)*K-Softmax

https://cs.stanford.edu/people/karpathy/connetjs/demo/cifar10.html

https://cs.stanford.edu/people/karpathy/convnetjs/demo/image_regression.html

Deep Learning Demos

- https://playground.tensorflow.org/
- http://deeplearning.net/demos/