Neural Networks 2

Intro to ML 466/566

Fall 2022

Some of these slides are from my lecture at the Neuromatch Deep Learning school.
https://deeplearning.neuromatch.io/tutorials/intro.html

Administrivia

- As 1 due tonight
- 566 project feedback is on eclass
  - Let me know if you can’t see it
- 566 teams, please book a time to meet with me.
  - Link on eclass

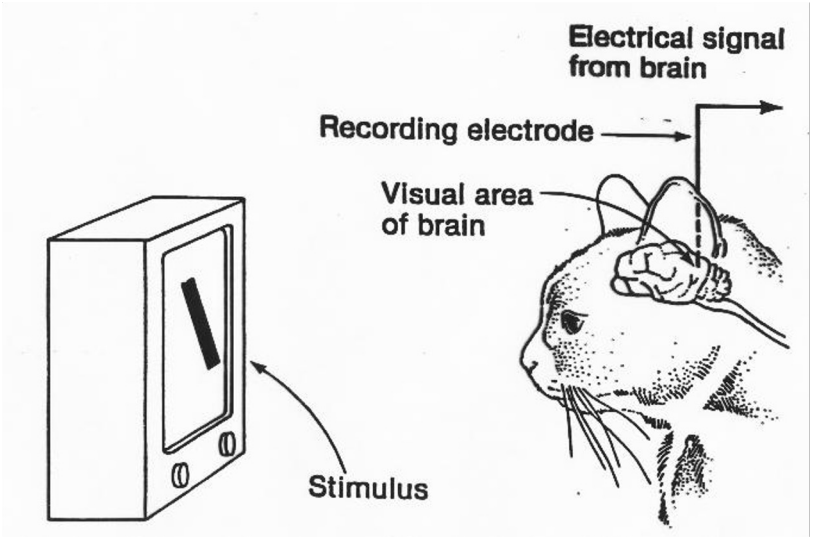
Invariances

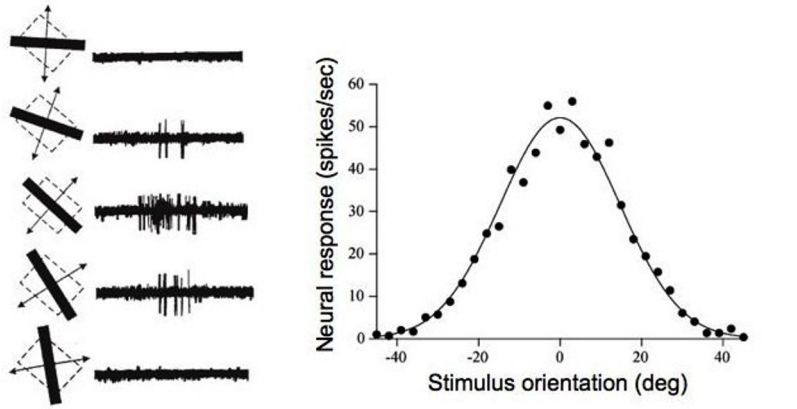
- Would like to capture the invariances we see in images
  - objects can appear at any place in the image

Visual Processing in the Brain

- The brain is an efficient machine
- How does it “solve” vision?

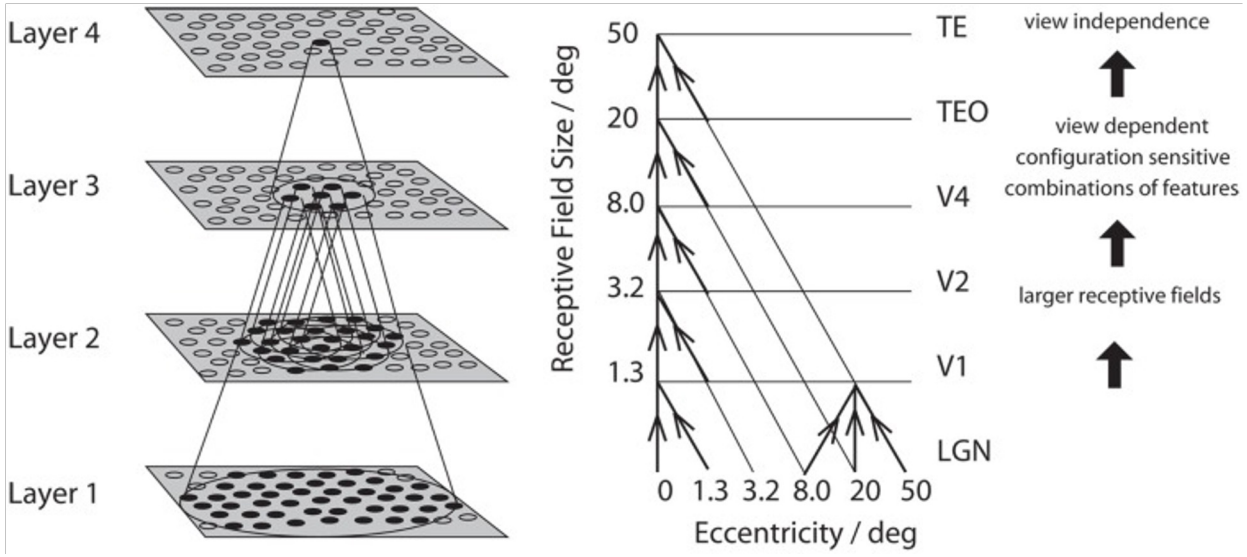
Hubel and Wiesel





Hubel & Wiesel, 1968

A hierarchy of processing



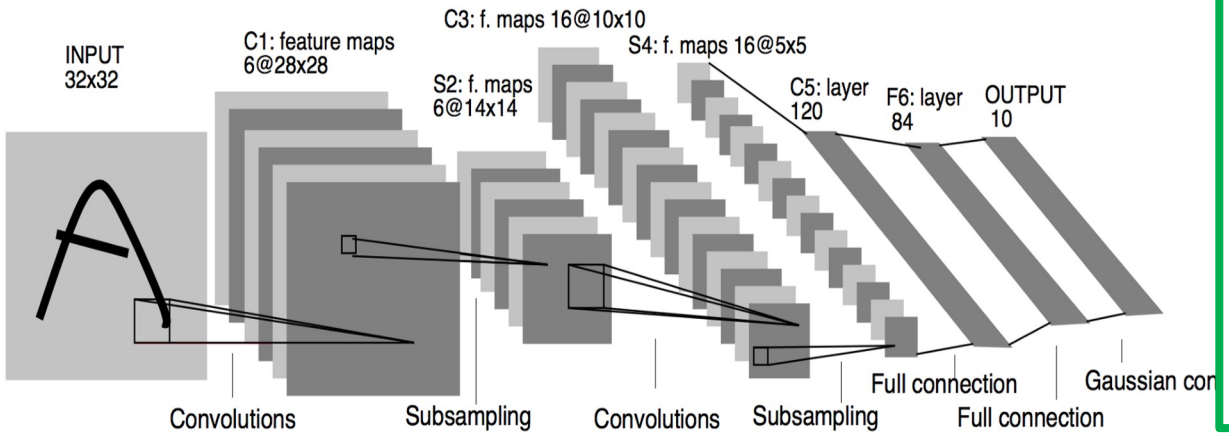
Rolls 2012

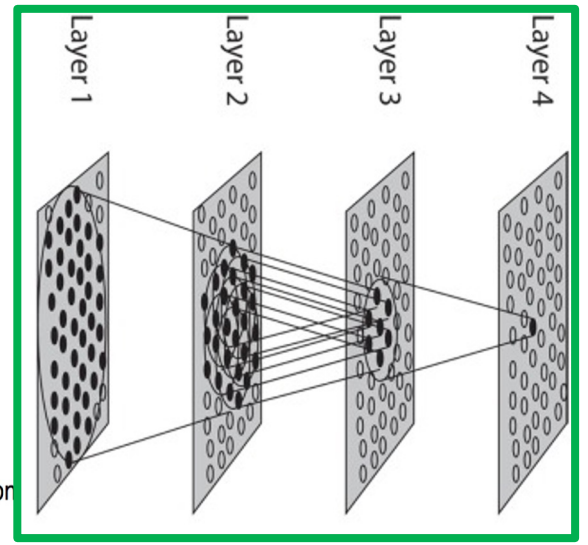
Invariances

Early history of CNNs

LeNet (1998) -- Architecture

- Developed by Yann LeCun @ Bell labs



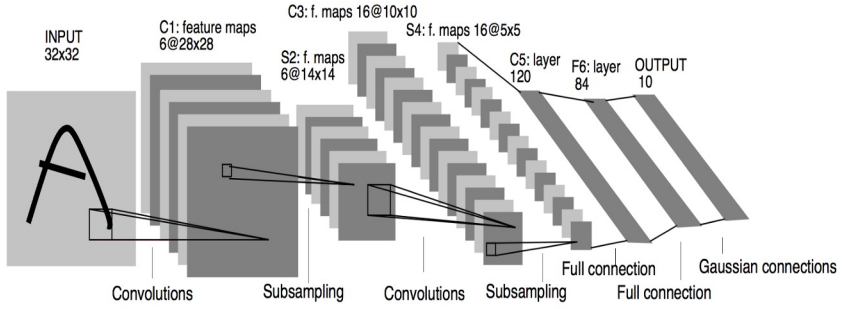


LeNet (1998) -- Results

- Successfully trained a 60K parameter neural network without GPU acceleration!
- Automatically reading zip codes on mail
- 0.8% error on MNIST; near state-of-the-art at the time.

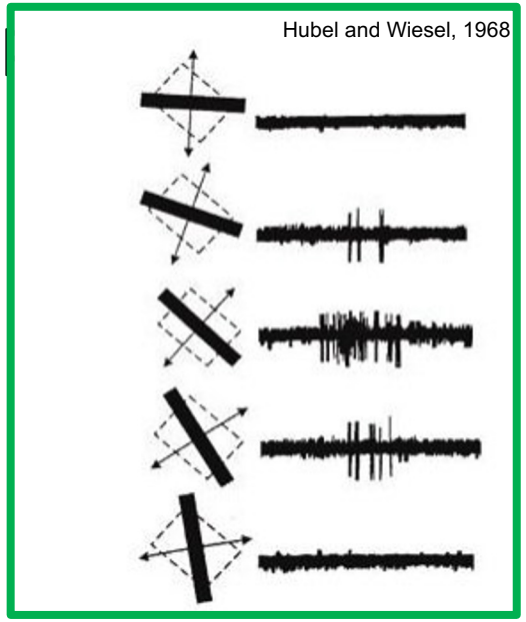
LeNet: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

CNNs: Convolution

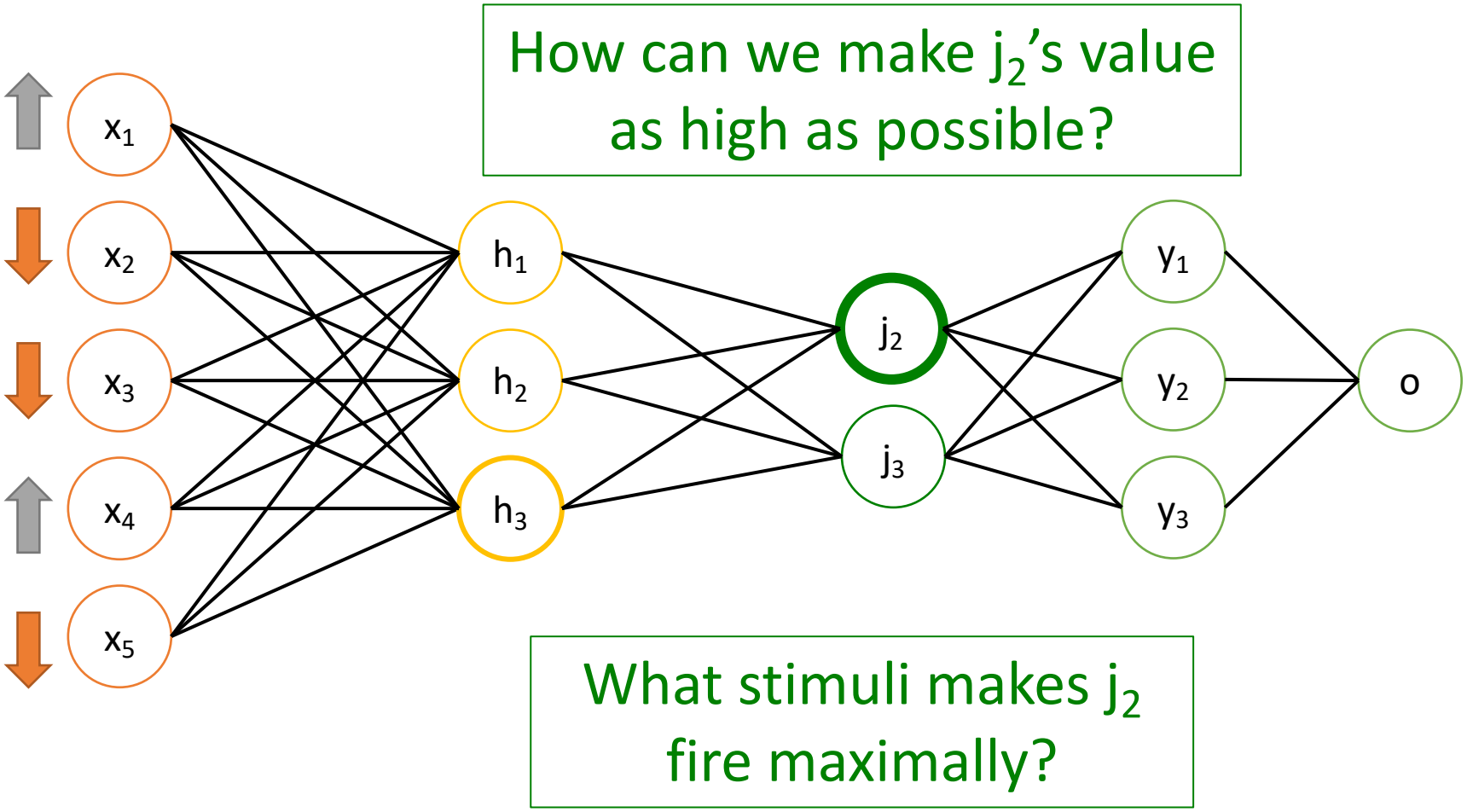


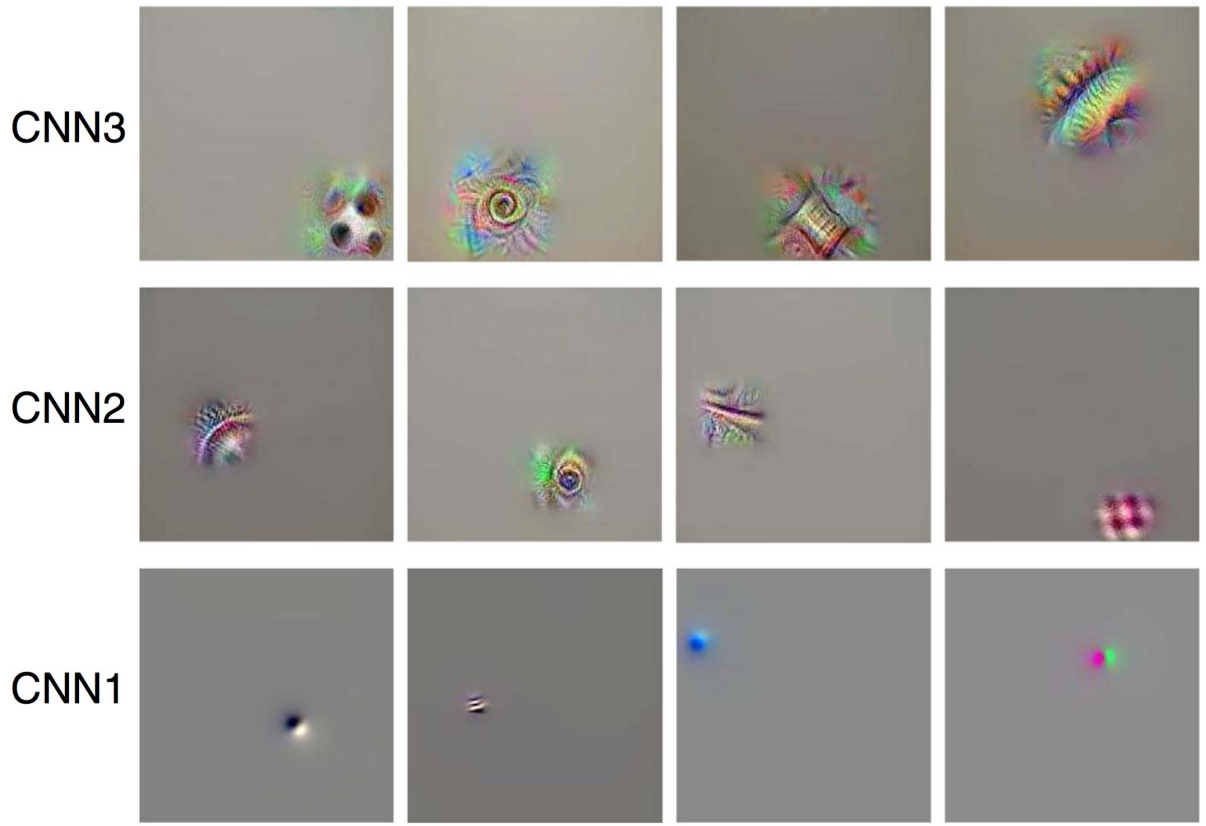
- Typical Learned Filters (C1 feature ma

Krizhevsky et al., 2012



What do the deeper neurons represent?





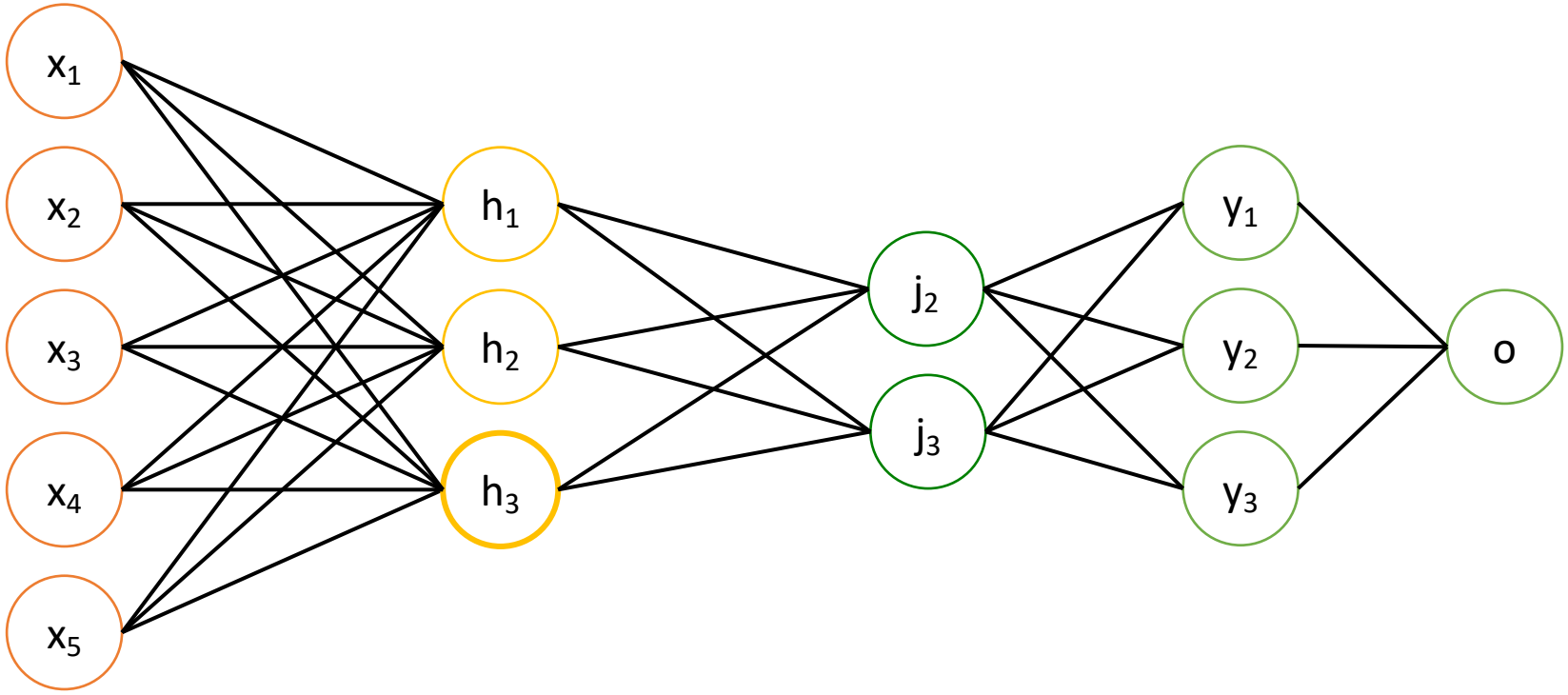
Horikawa & Kamitani (2017)

Horikawa & Kamitani (2017)

DL is a story of representation learning

Hence ICLR

What is a representation?



DL is a story of representation learning

What is a representation?

A symbol or construct that approximates the entities and/or relations in the real world

Your brain has representations of the real world because it cannot actually *contain* the real world

DL is a story of representation learning

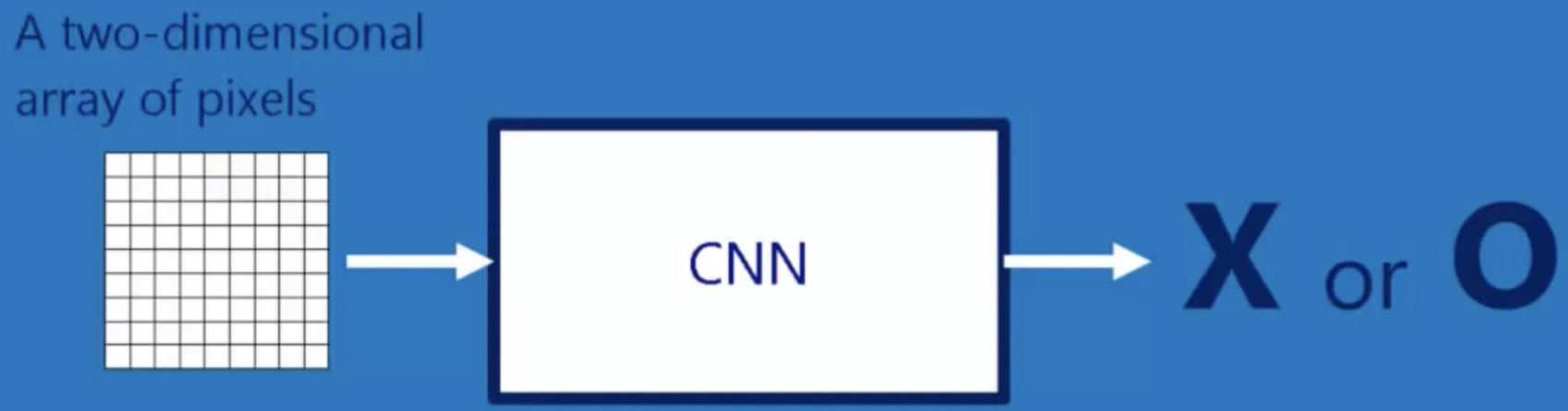
Your brain's representations allow it to approximate the real world in very useful ways.

E.g. you understand that some objects are more similar than others, you understand that objects can work together, objects can be parts of other objects, etc…

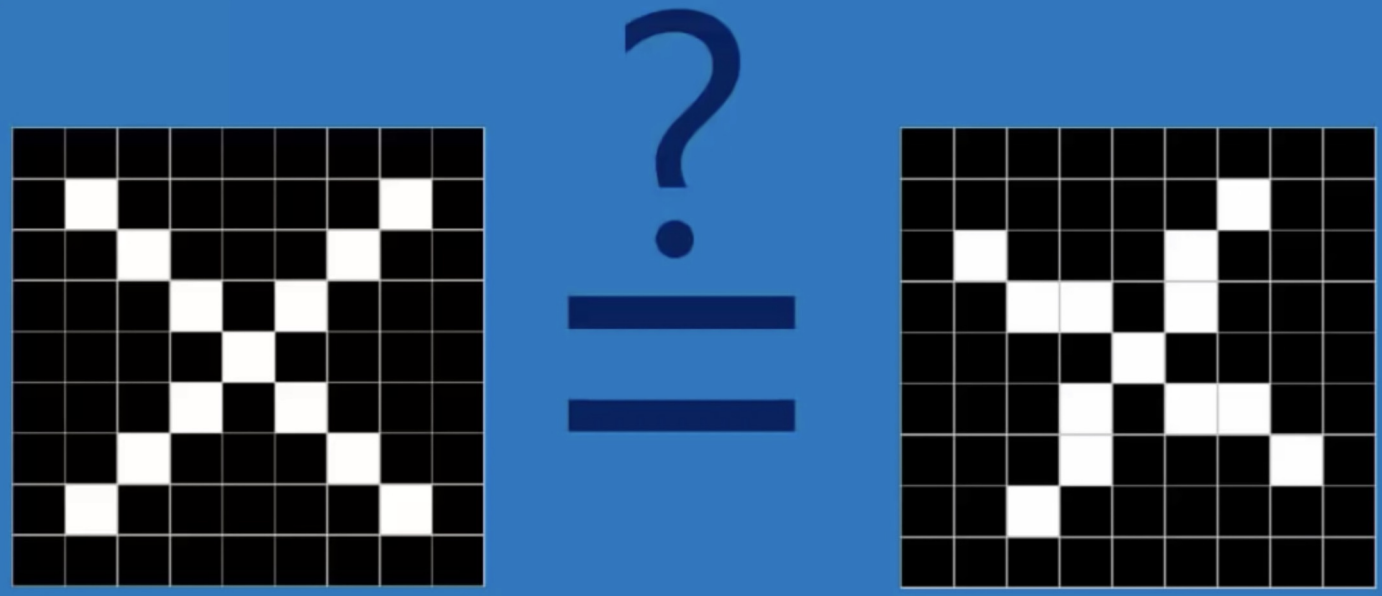
What do we want a good representation to do for us?

What is a CNN?

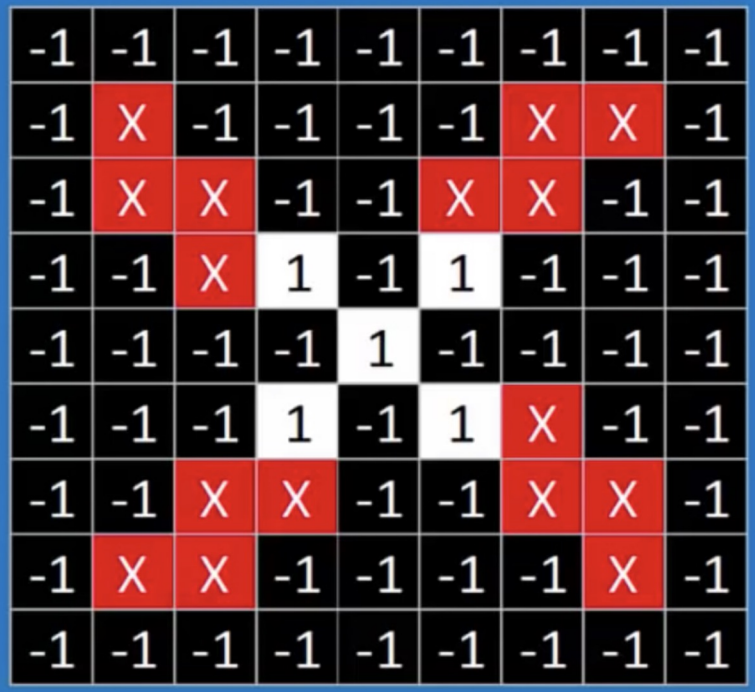
All the (blue) slides adapted from Brandon Rohrer with permission



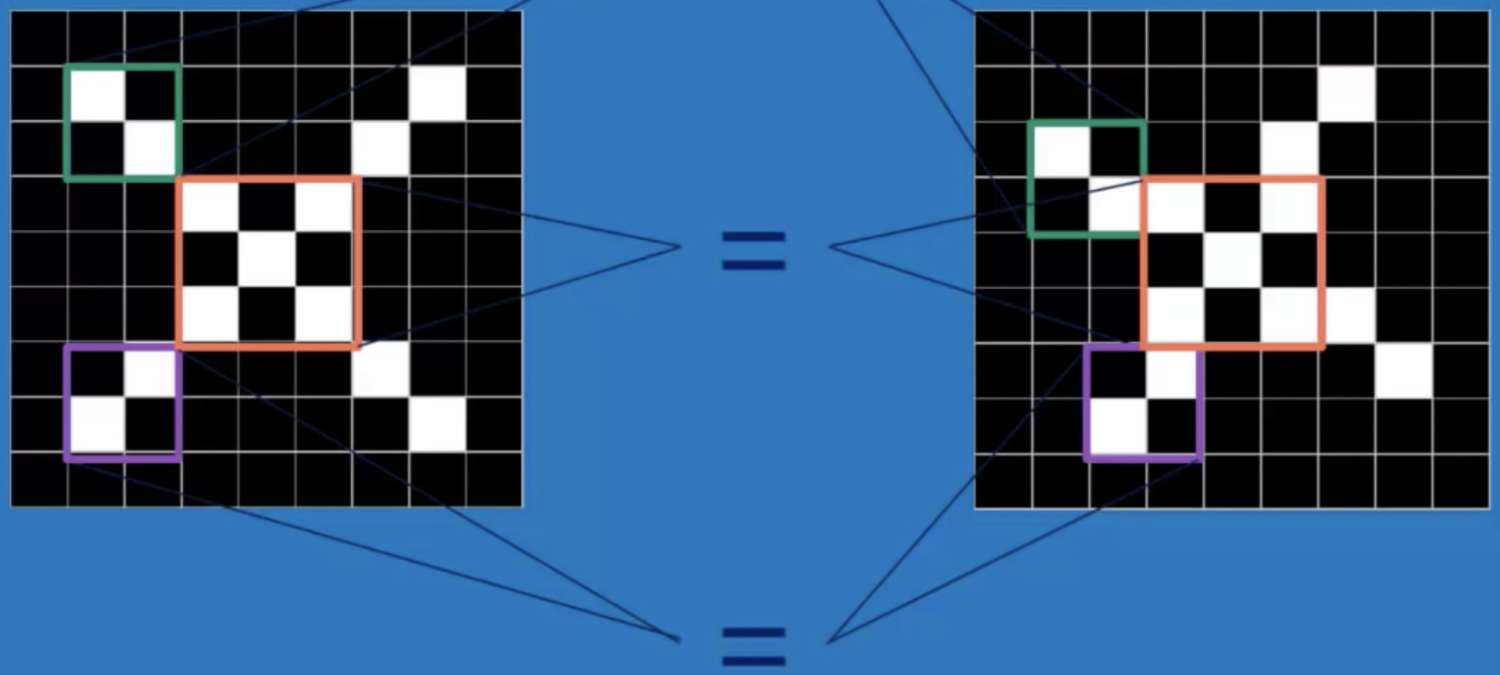
Nontrivial



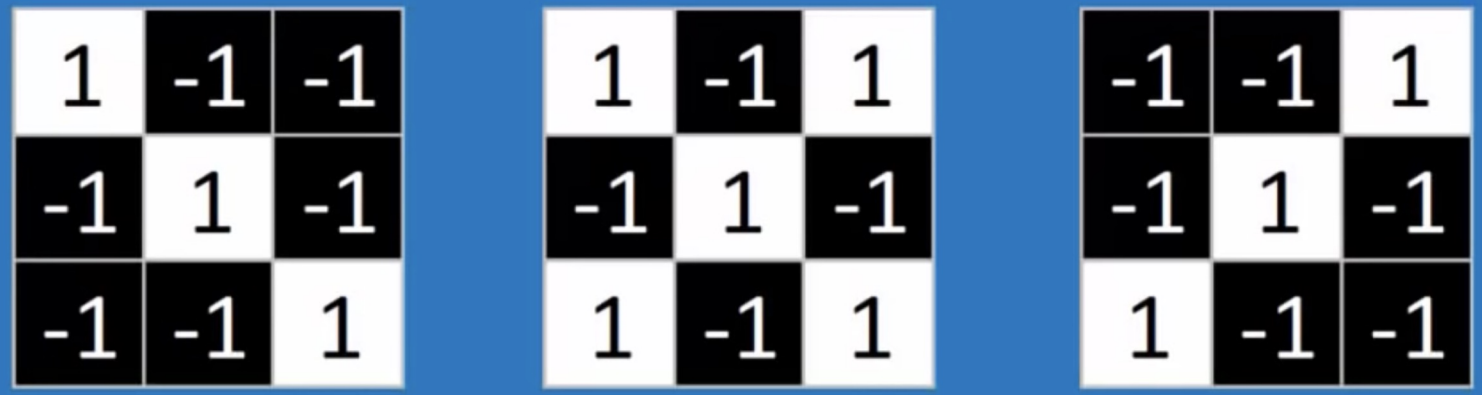
Naive comparison



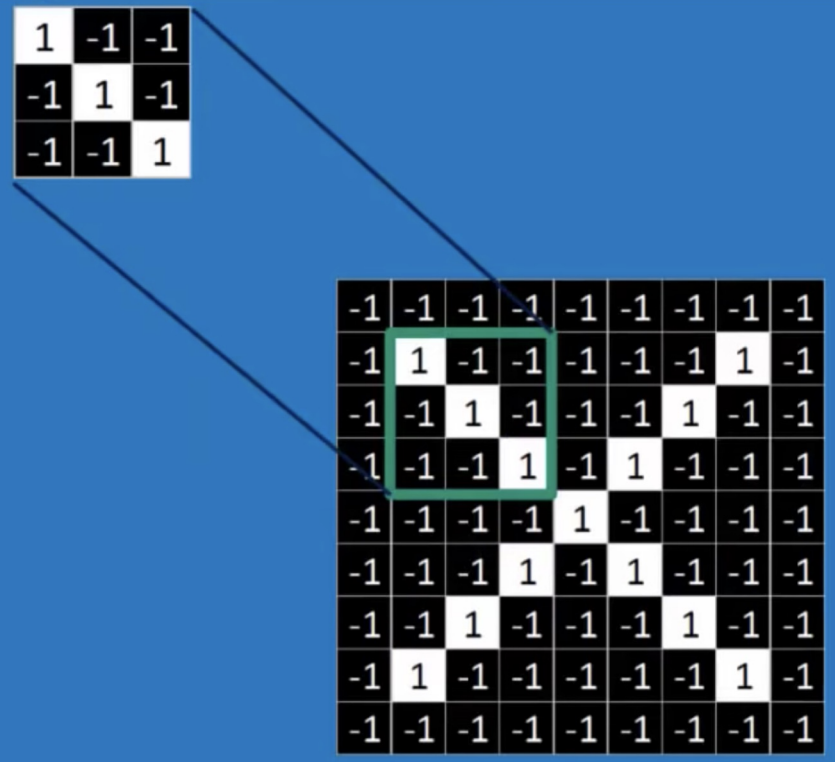
Intuition



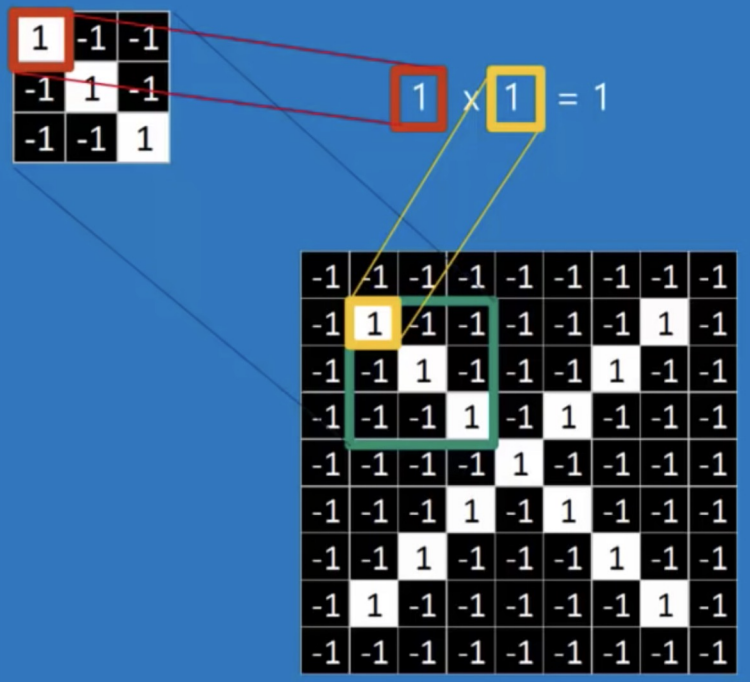
Potential local features



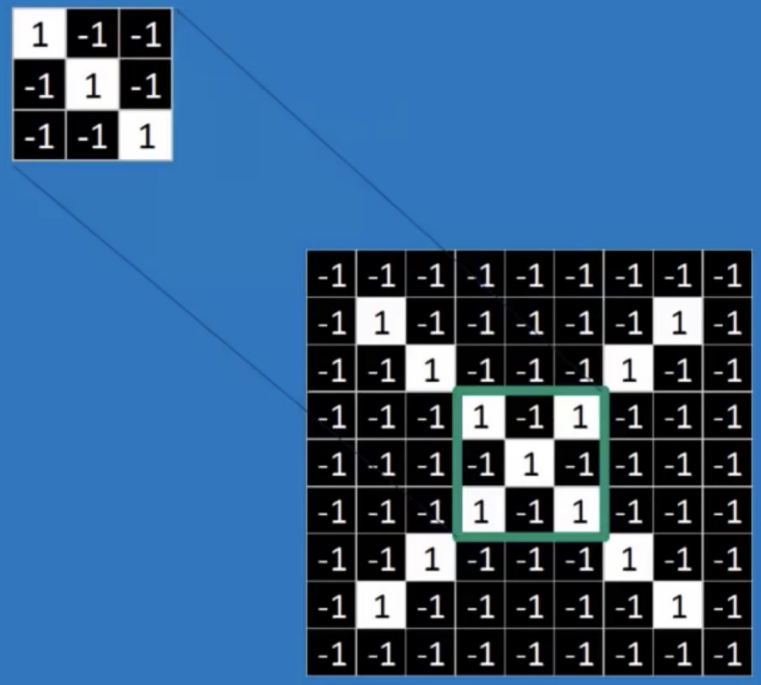
Setting


1 -1 -1
-1 1 -1
-1 -1 1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1

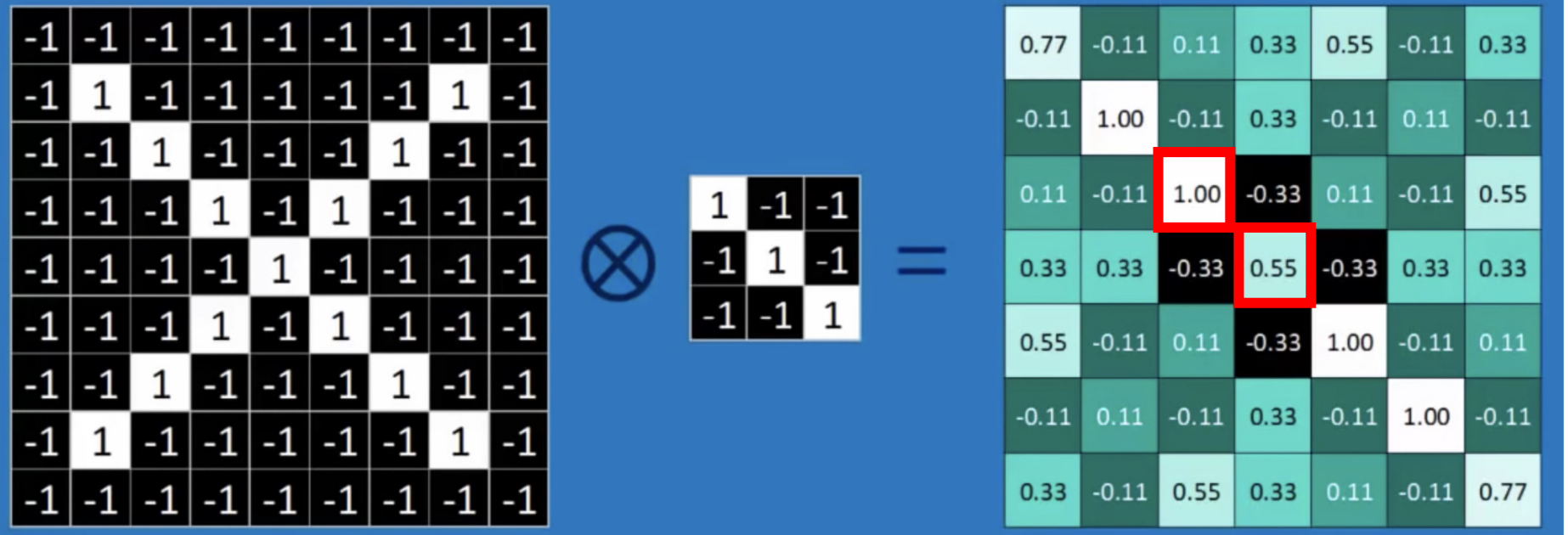
Local filtering



And at different locations



Convolution



This is 2D convolution

- Convolution can also be 1D and 3D
- Convolutions can be used in other application areas
  - Whenever there’s predictable correlation over time/space
  - Language, protein or DNA sequences

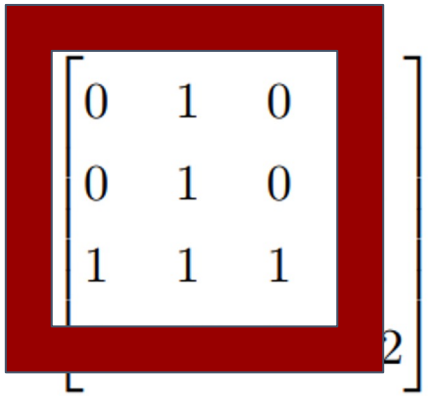
A convolution exercise

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & -2 & 0 & -2 \end{bmatrix} \qquad \begin{bmatrix} -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \end{bmatrix}$$

Here's our "image"

Here's our "filter"

A convolution exercise



$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \end{bmatrix}$$

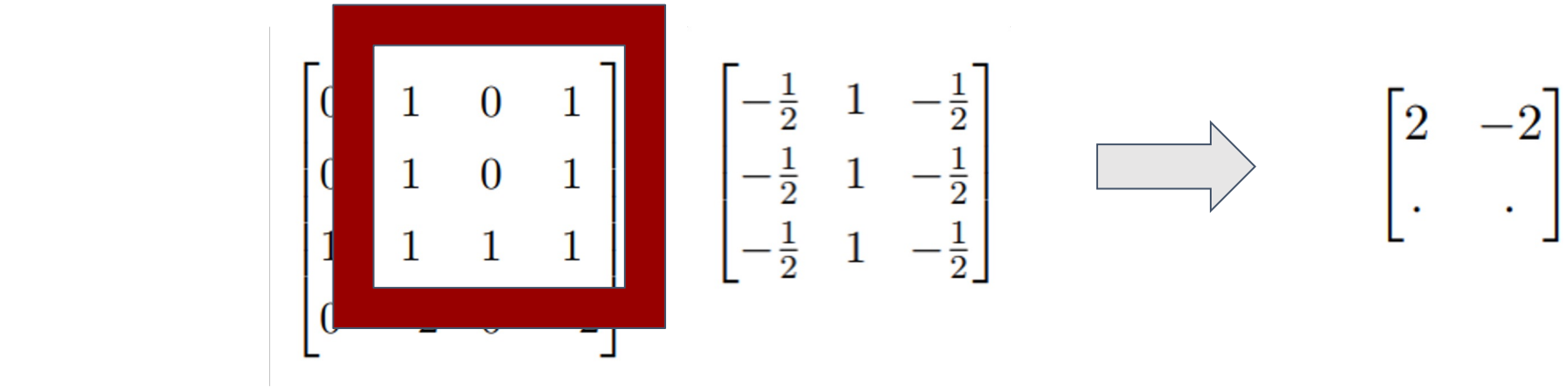
$$\begin{bmatrix} 2 & \cdot \\ \cdot & \cdot \end{bmatrix}$$

$$\left(0 \times \frac{-1}{2}\right) + (1 \times 1) + \left(0 \times \frac{-1}{2}\right)$$

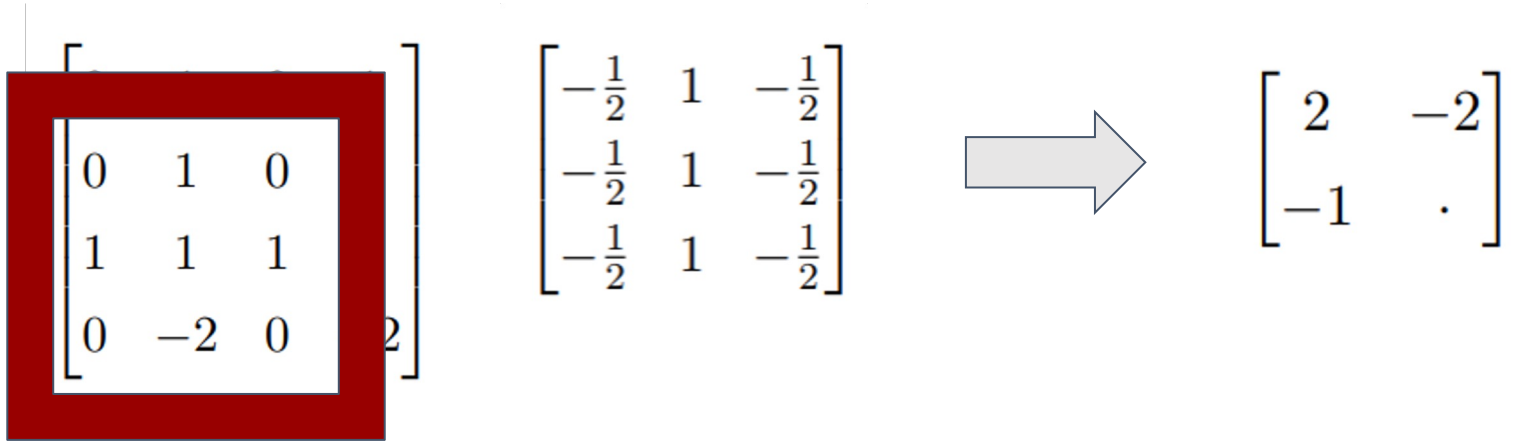
$$\left(0 \times \frac{-1}{2}\right) + (1 \times 1) + \left(0 \times \frac{-1}{2}\right)$$

$$\left(1 \times \frac{-1}{2}\right) + (1 \times 1) + \left(1 \times \frac{-1}{2}\right) = 2$$

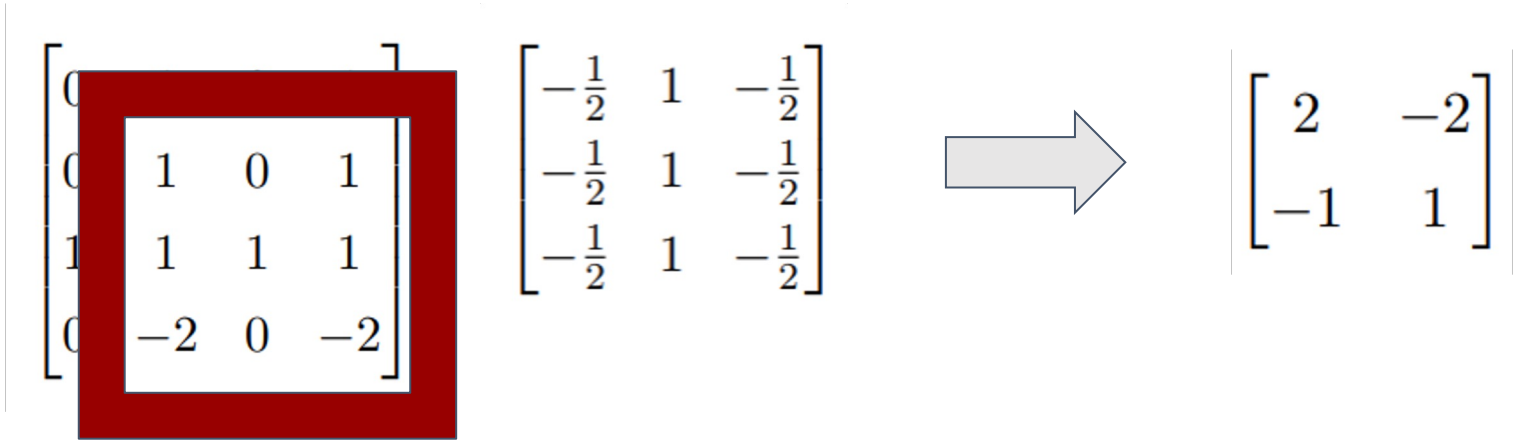
A convolution exercise



A convolution exercise

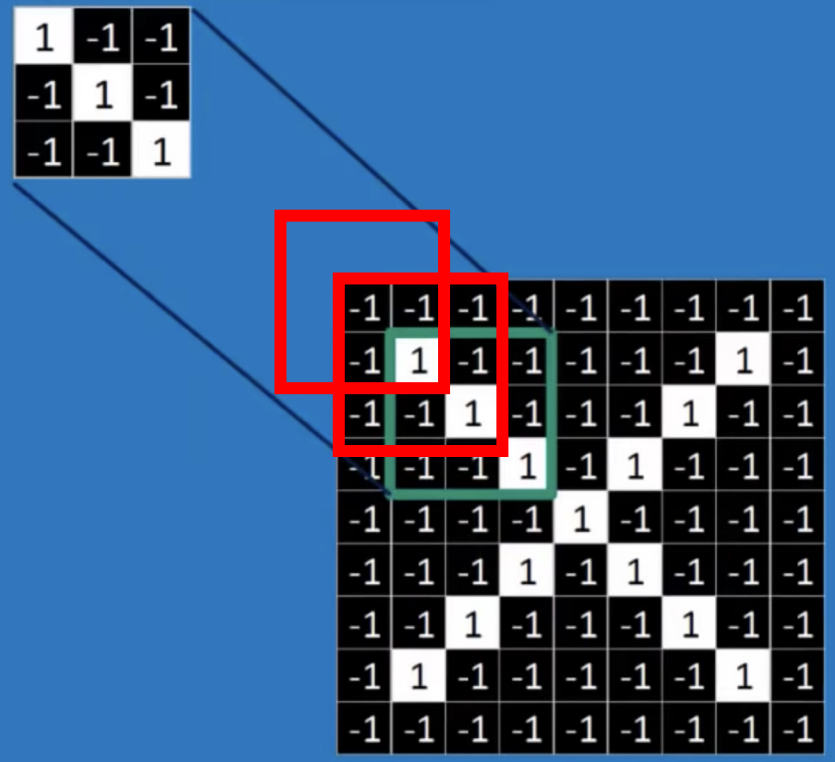


A convolution exercise

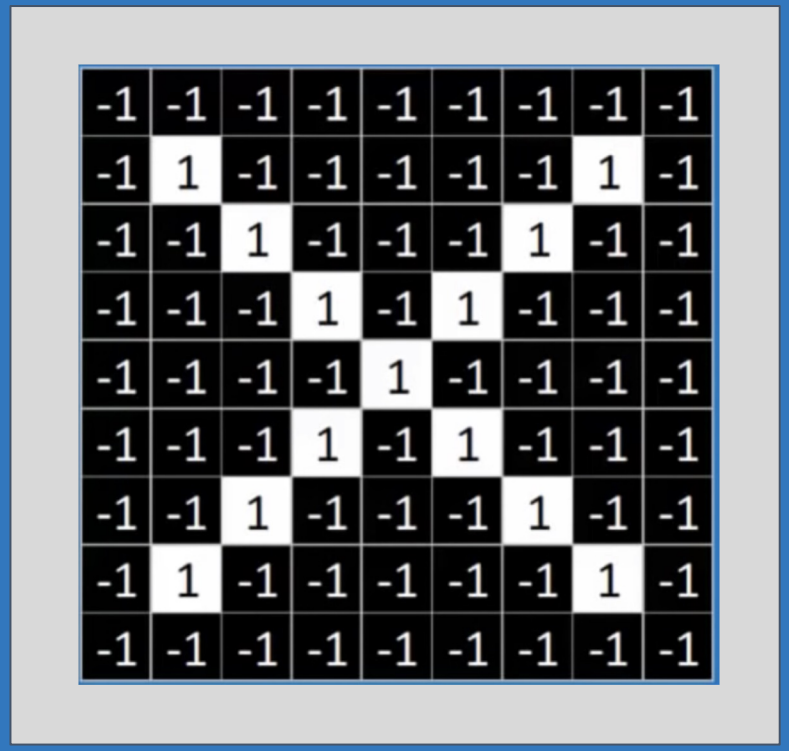


What about the edges of an image?

Padding



Padding


-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1

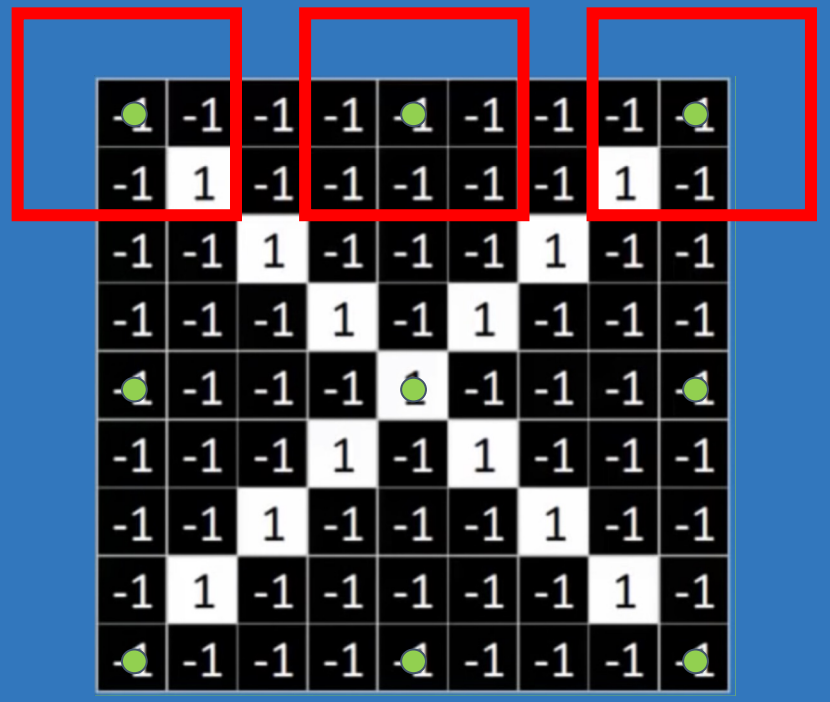
Stride 2


-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1

Stride 3


-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 -1 -1 1 -1 -1 -1
-1 -1 -1 -1 1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1

Stride 4


-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 1 -1 1 -1 -1 -1
-1 -1 1 -1 -1 -1 1 -1 -1
-1 1 -1 -1 -1 -1 -1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1

Motivation: conv for edge detection

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal lines

-1	2	-1
-1	2	-1
-1	2	-1

Vertical lines

-1	-1	2
-1	2	-1
2	-1	-1

45 degree lines

2	-1	-1
-1	2	-1
-1	-1	2

135 degree lines

Horizontal lines:

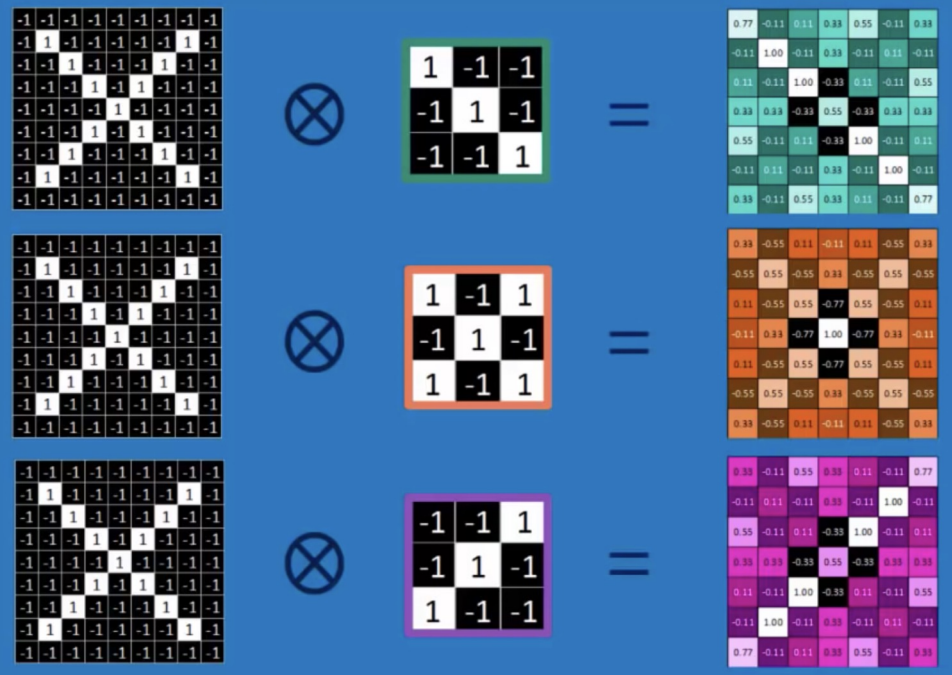
Feature detection - edges

Filters give us **global** translation invariance

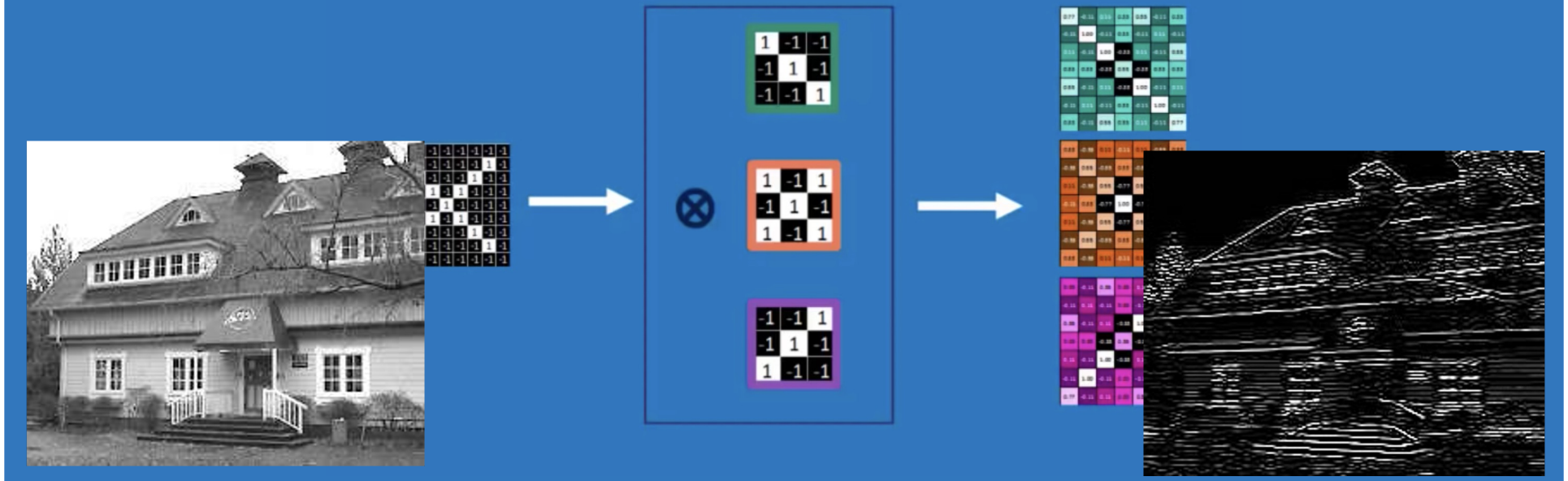
Run a filter across an image

How could that be useful for image recognition?

Multiple filters

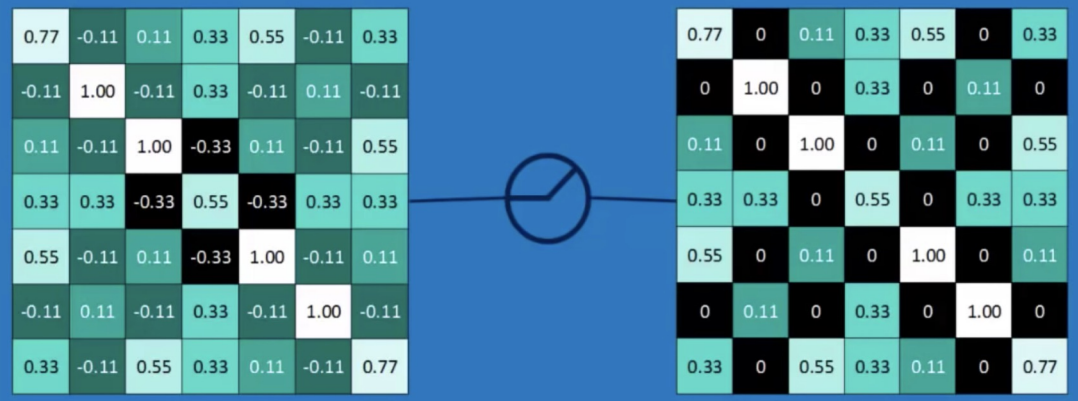


A convolution layer



Potential to add a ReLU

ReLU(x) = max(0,x)



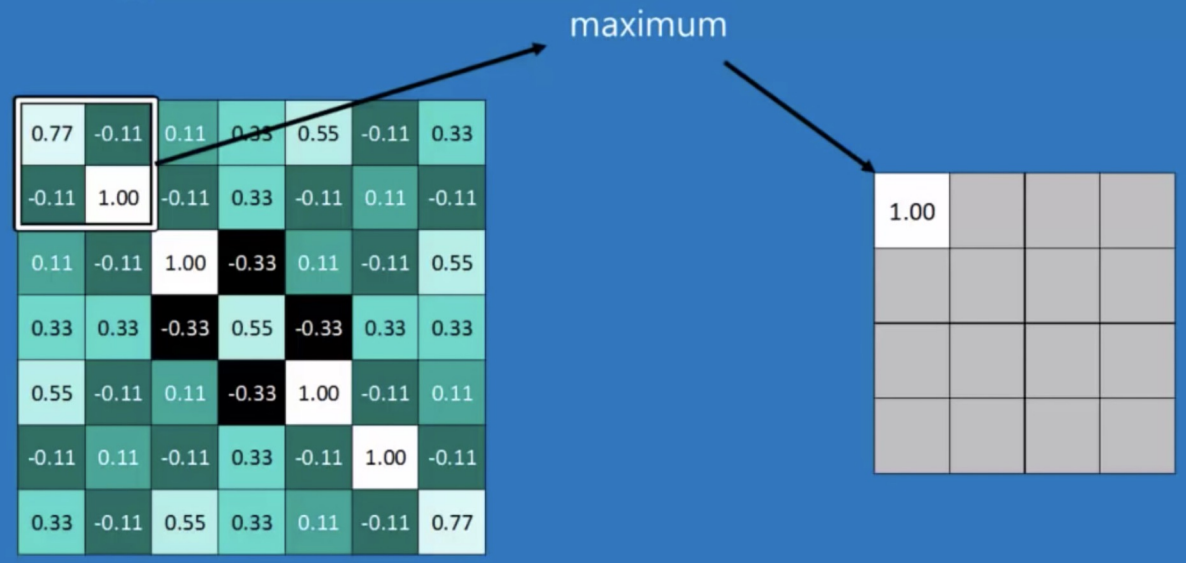
Pooling = Local invariance

- Recall: filters give us global invariance
- Pooling gives local invariance

The need for some invariance

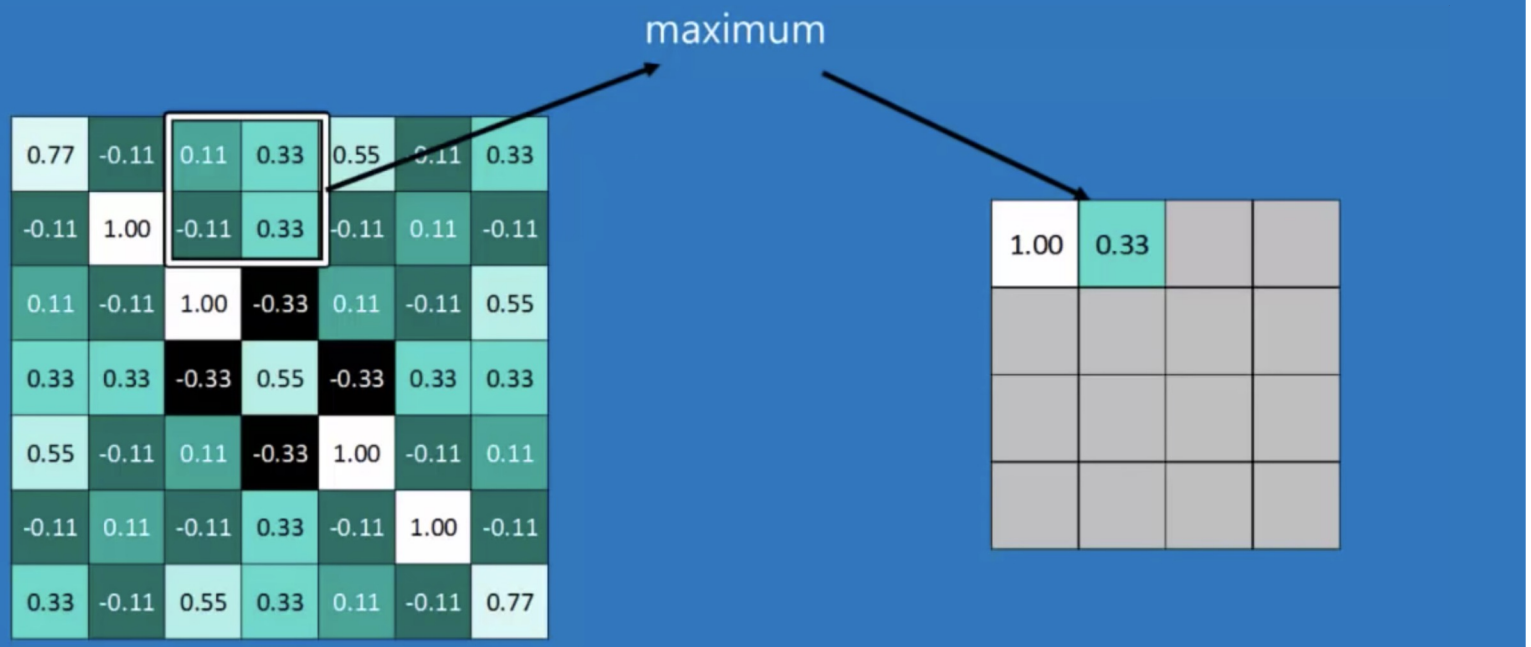
Features may appear in (slightly) different places

Max-pooling

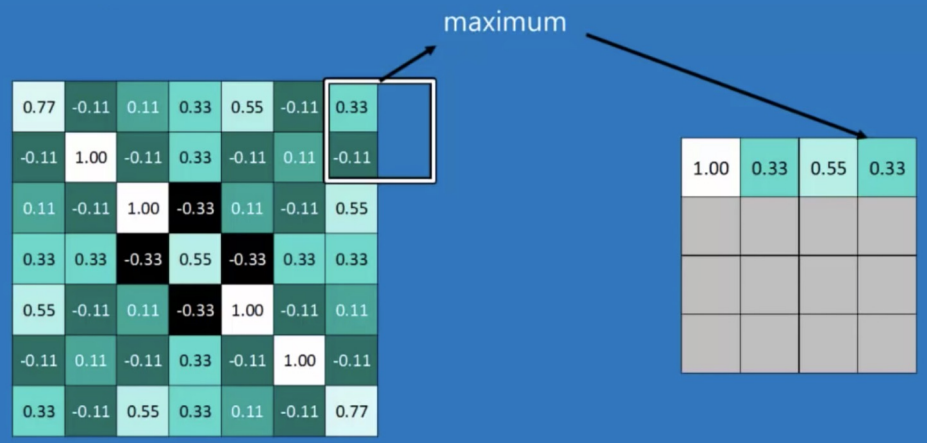


2*2, Stride 2

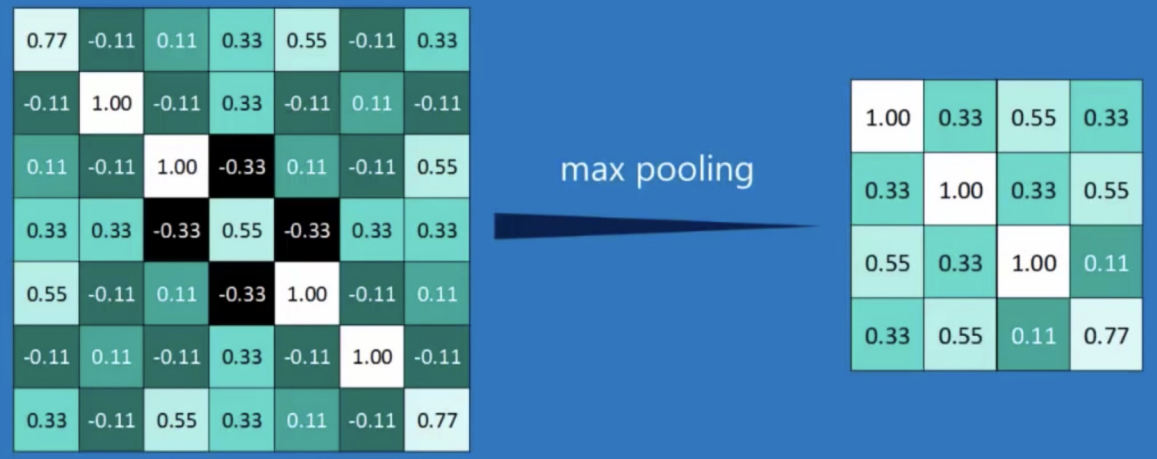
Across space



Again, Padding problem!

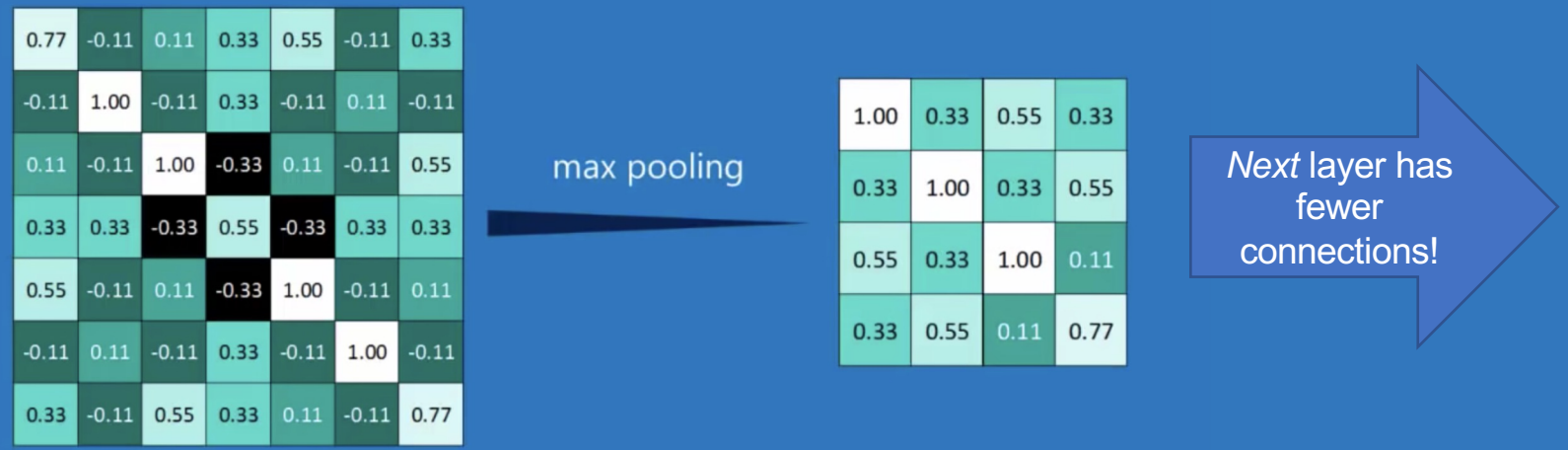


And a whole max pooling operation

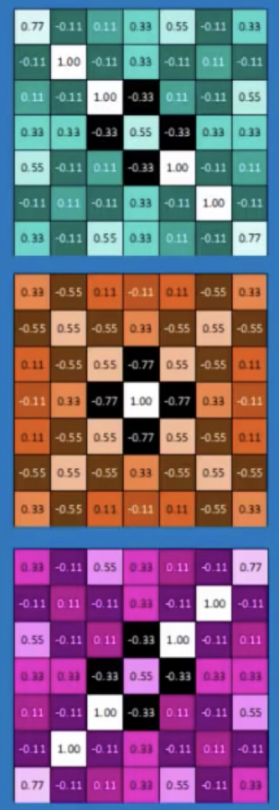


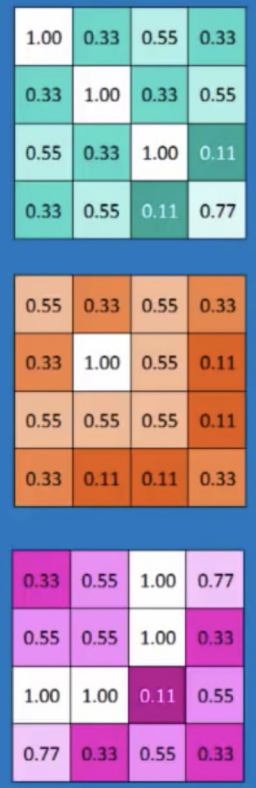
How many parameters?

Hold up, how does this reduce the # of params?



Bunch of Filters



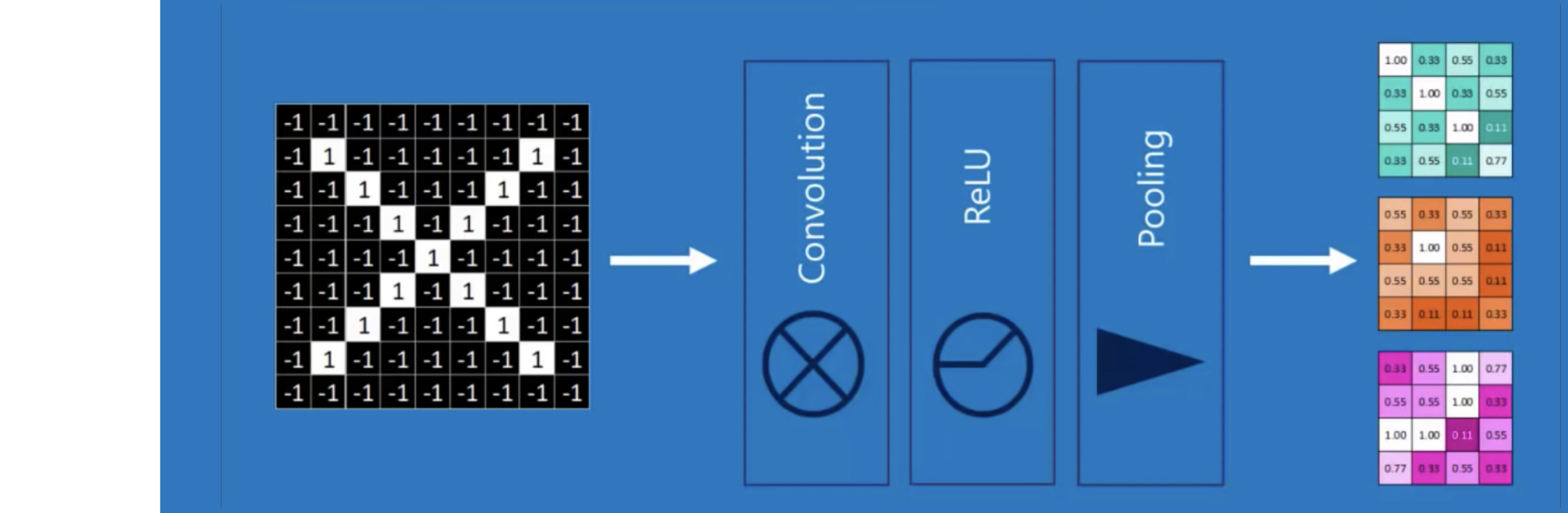


Pooling layers

- Want to:
  - Reduce dimensions of data, without additional parameters
  - Introduce some ***local*** translation invariance
- Pooling operation can be max(0,x) or average(x)

Putting it all together

So far: not much of a hierarchy



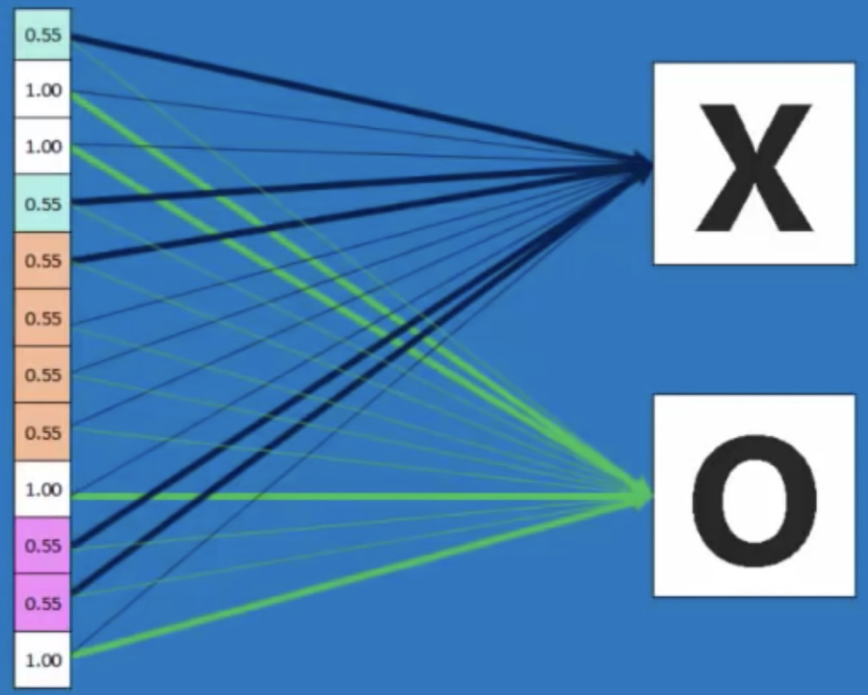
Chain it


Convolution
ReLU
Convolution
ReLU
Pooling
Convolution
ReLU
Pooling
1.00 0.55
0.55 1.00
1.00 0.55
0.55 0.55
0.55 1.00
1.00 0.55

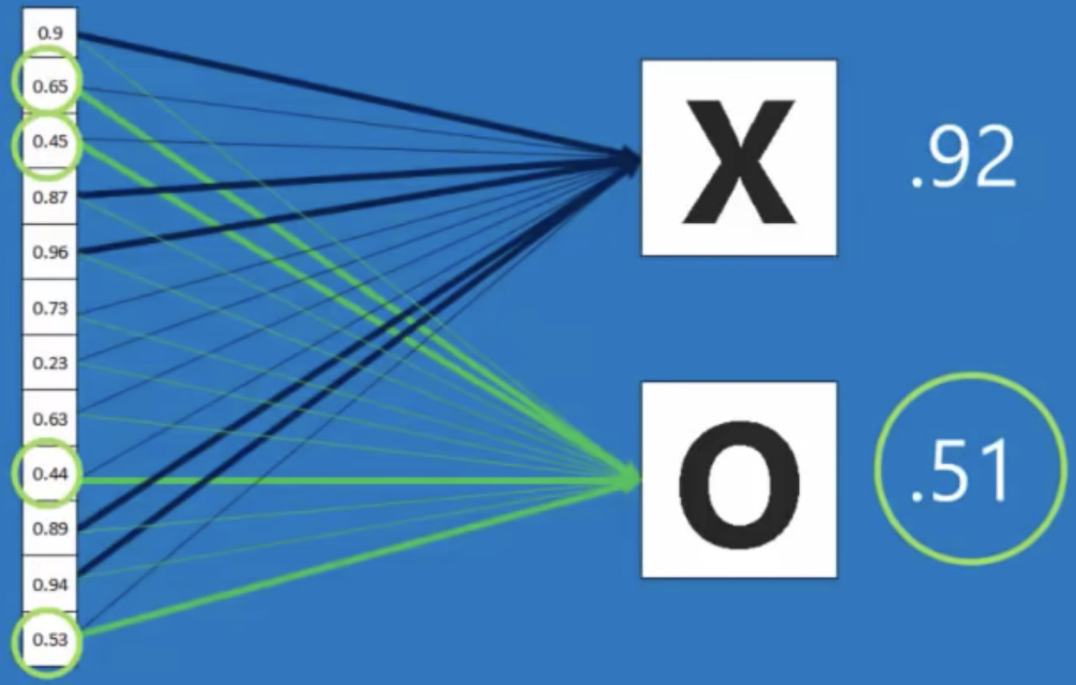
How to transition to fully connected

Fully connected read out

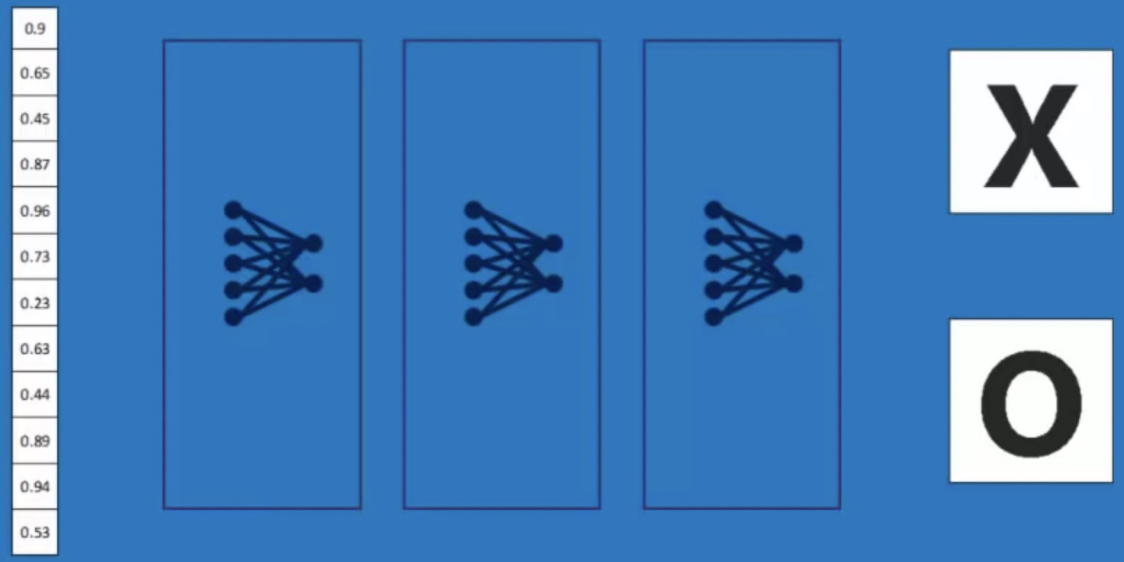
Predict both



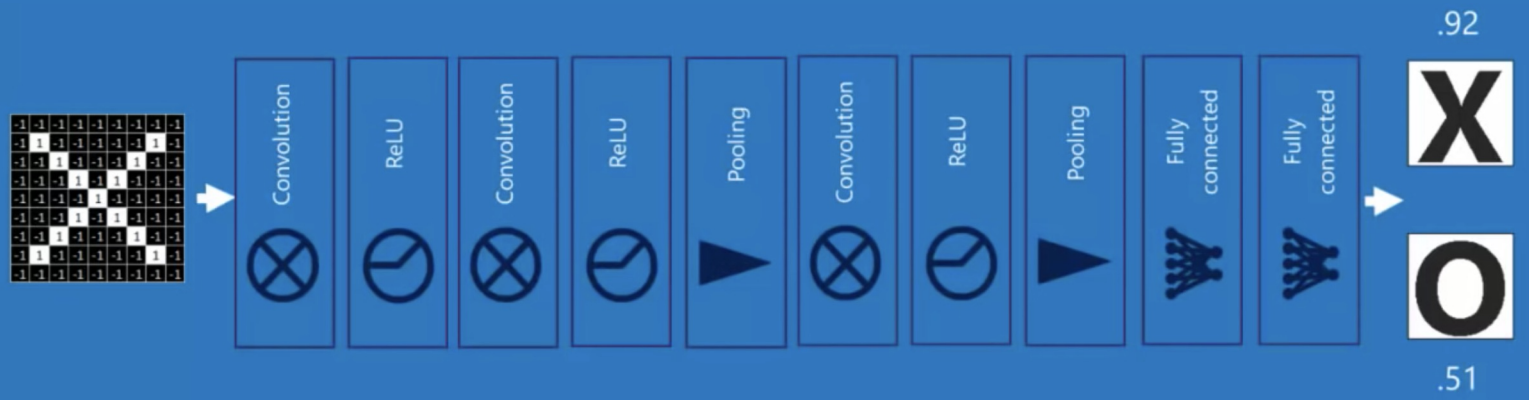
Get activations



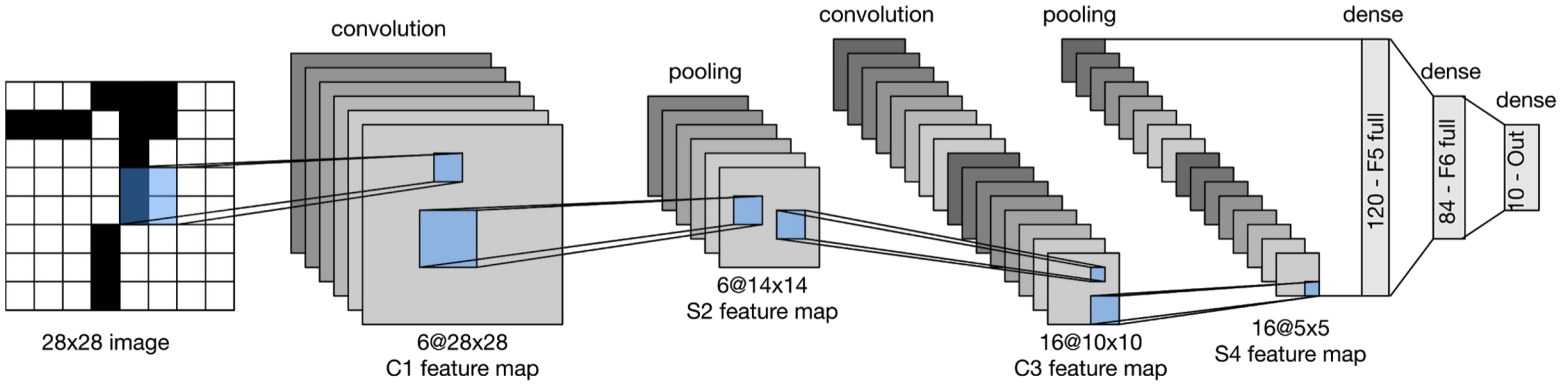
Multiple layers of fully connected



Now stack it all together



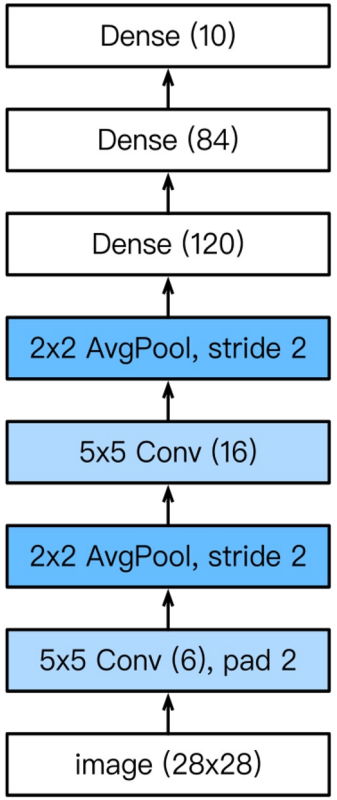
Putting things together



Input→(Conv→ReLU→Pooling)→...→Fully connected

From ‘Dive into deep learning’

Another way of visualizing the same network

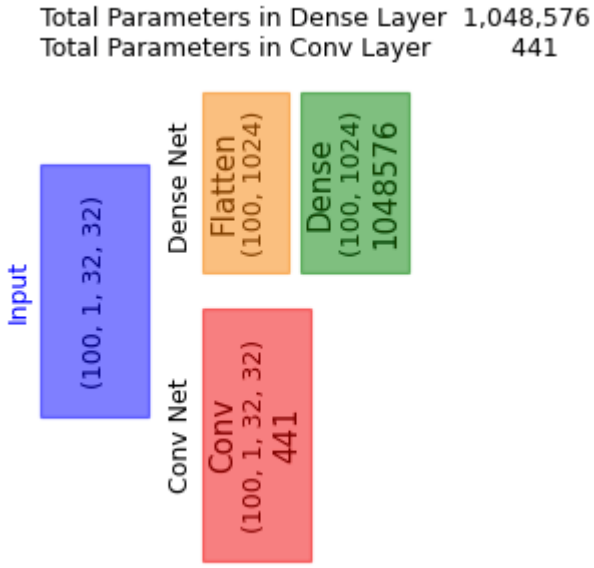


Let us calculate that: Convolution and pooling

- 1*1 image with 2 padding is 5*5
- 5*5 image, 5*5 filter => 1 output
- 300*400 image with 2 padding is 304*404
- 304*404 image, 5*5 filter =>300*400
- Maxpool 2x2 pooling window 300*400, stride 2 =150*200

Dense vs conv layers

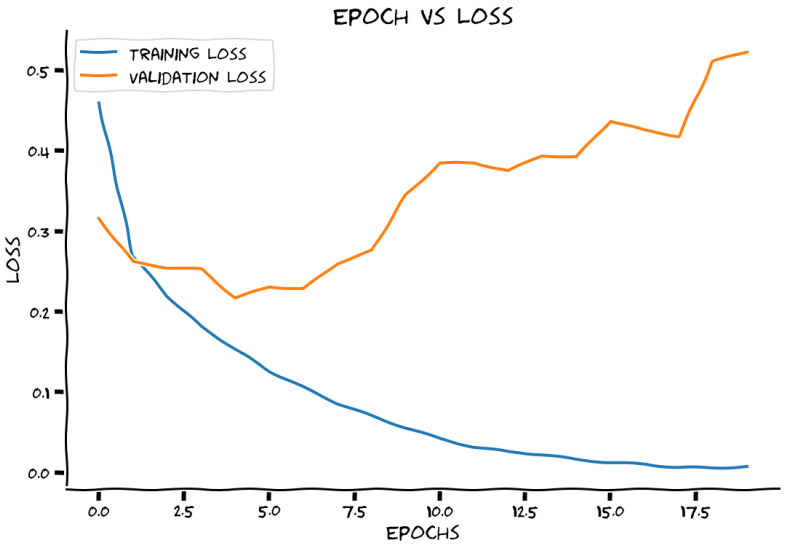
- Let’s compare the number of parameters in networks that use dense (fully connected) vs convolutional layers

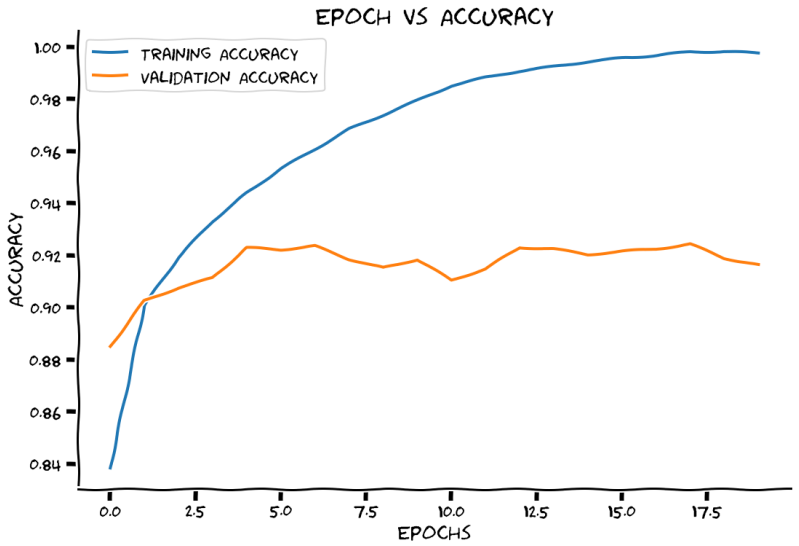


https://colab.research.google.com/drive/1_YP5SicR49OrlWpct0IrrAI5r4LsTDVz?usp=sharing

Overfitting

Below are strong signs of overfitting.





What can we do? Dropout

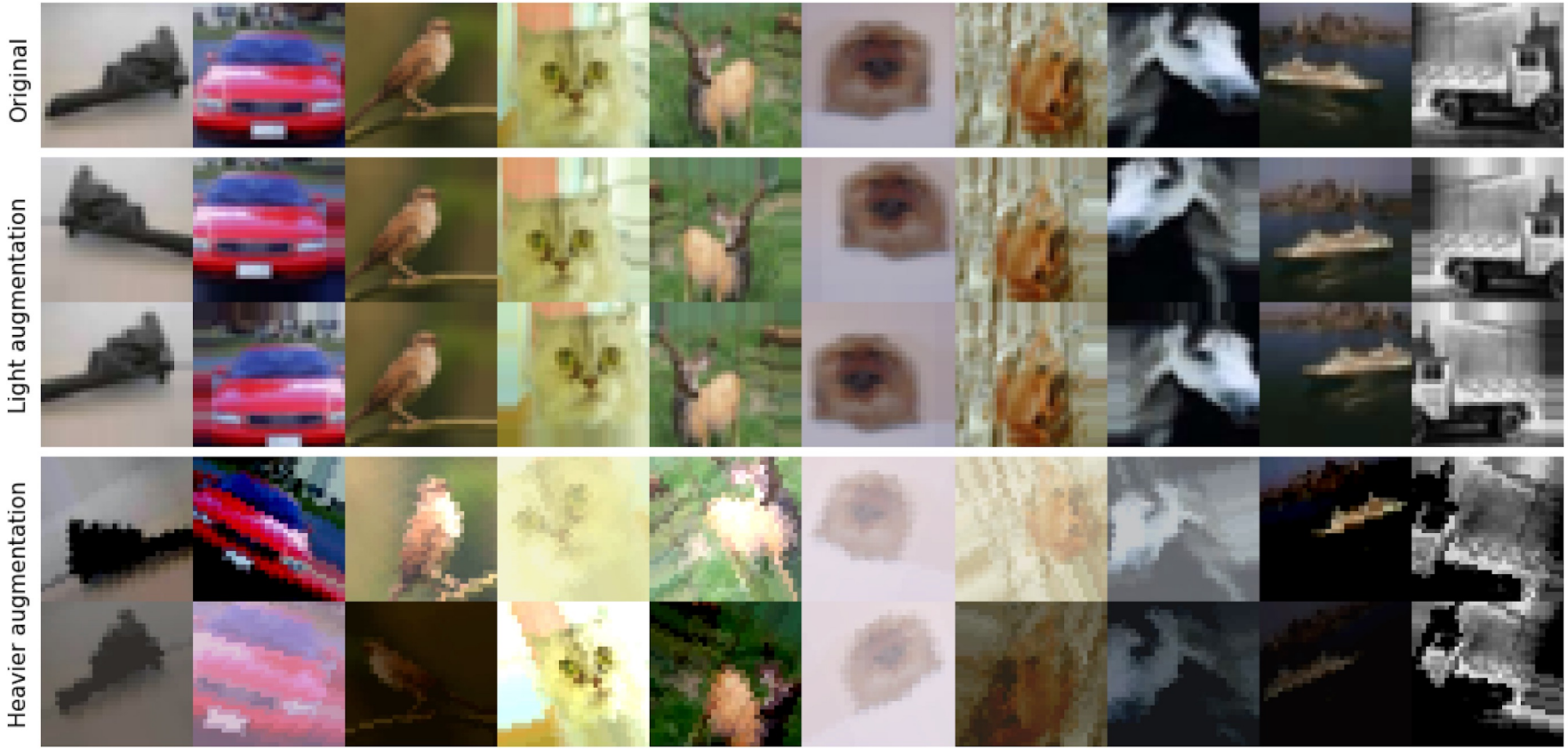
Intuition: no one neuron should be *too* important
Just like the brain, there should be redundancy in connections

Dropout: randomly set some of the hidden activations to 0
Parameter: % of neurons to set to 0

What can we do? Data augmentation (DA)

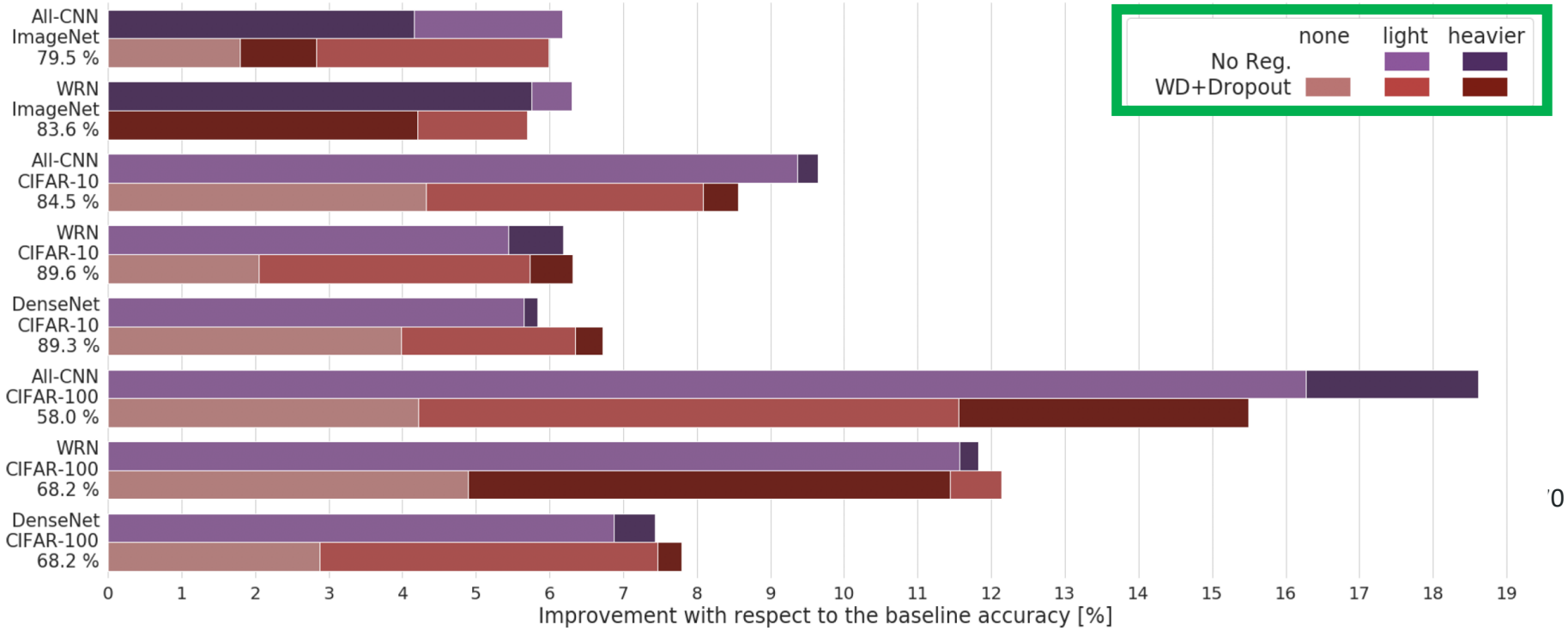
source: Hernandez Garcia Thesis

Different strengths of DA



source: Hernandez Garcia, arxiv
https://arxiv.org/pdf/1806.03852.pdf

Data augmentation is often more important than direct regularization



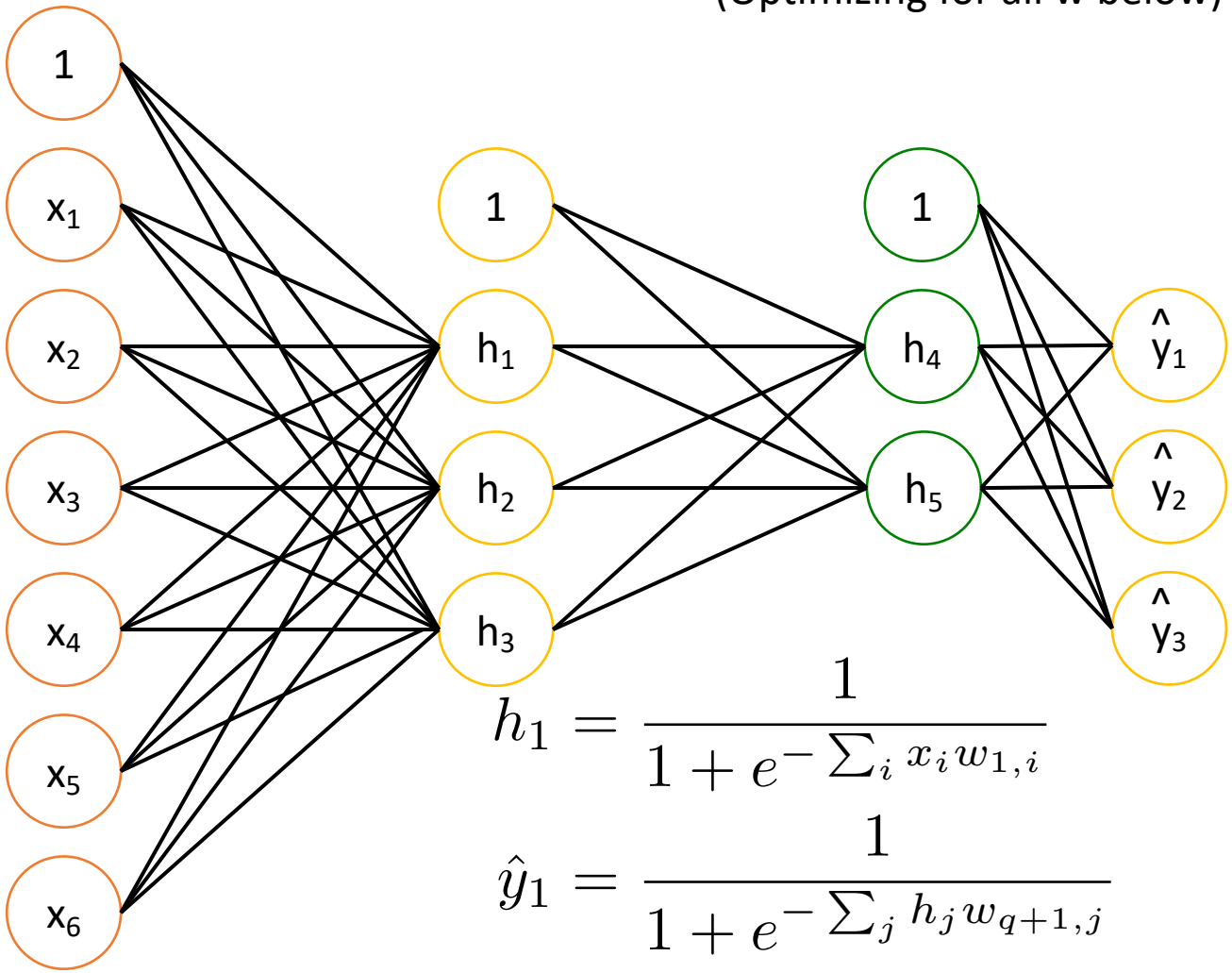
source: Hernandez Garcia, arxiv

Interpretability

- Chris Olah has a great series on interpretability (the distill ones are particularly good)
  - https://colah.github.io/

Back Prop

(Optimizing for all w below)



$$h_1 = \frac{1}{1 + e^{-\sum_i x_i w_{1,i}}}$$

$$\hat{y}_1 = \frac{1}{1 + e^{-\sum_j h_j w_{q+1,j}}}$$

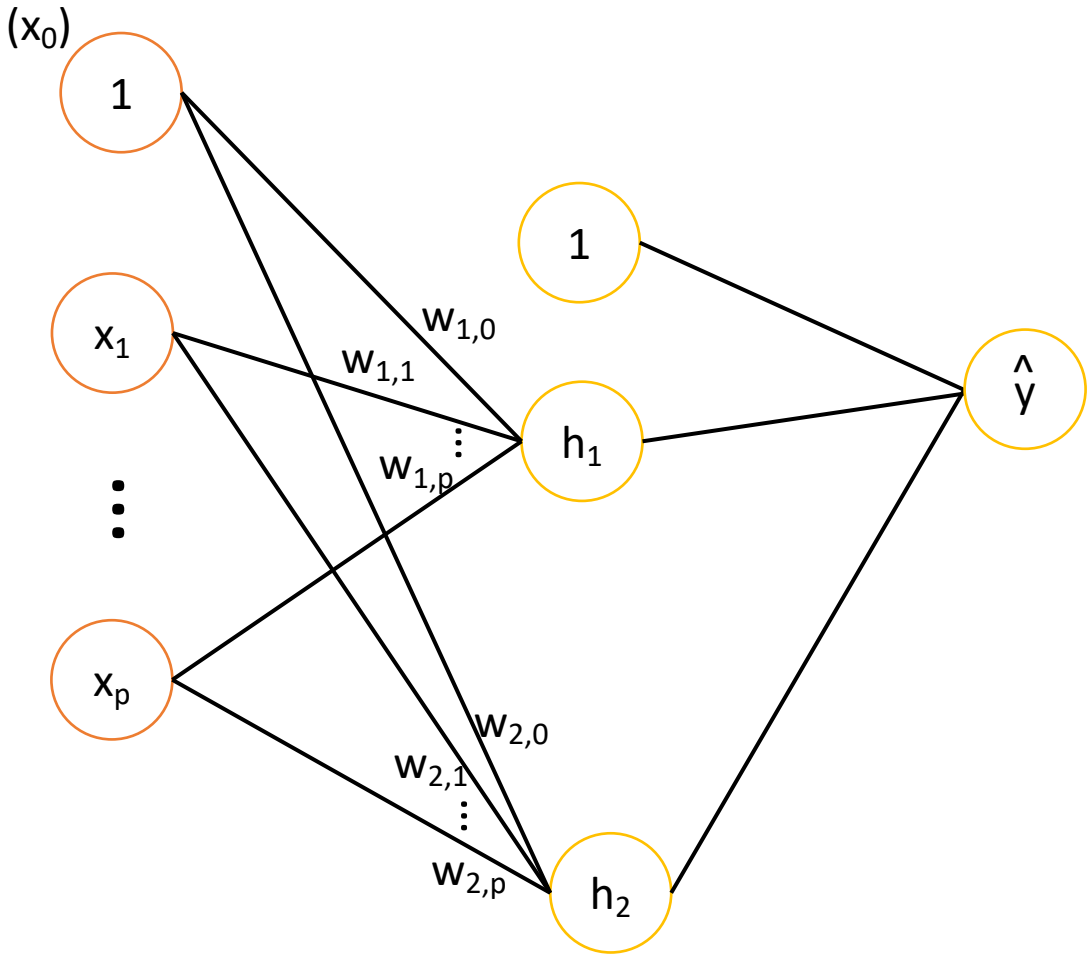
q is total # of hidden nodes across all layers

Backprop Algorithm

1. Randomly initialize weights (w)
2. Repeat until convergence
    1. For each (batch, mini-batch) in data
        1. For each data point in batch
            1. Forward pass (calculate all intermediate values h, s)
            2. Backwards pass compute gradient of loss wrt all w
        2. Average gradient over data points in batch
        3. Update w

Backprop takes advantage of the fact that many of the values you need for each gradient are computed in the forward pass, or as part of another gradient.

Our example for in class



$$h_1 = \frac{1}{1 + e^{-\sum_i x_i w_{1,i}}}$$

$$\hat{y} = \sum_j h_j w_{3,j}$$

(Optimizing for all w below)

$$\hat{y} = \sum_j h_j w_{3,j}$$

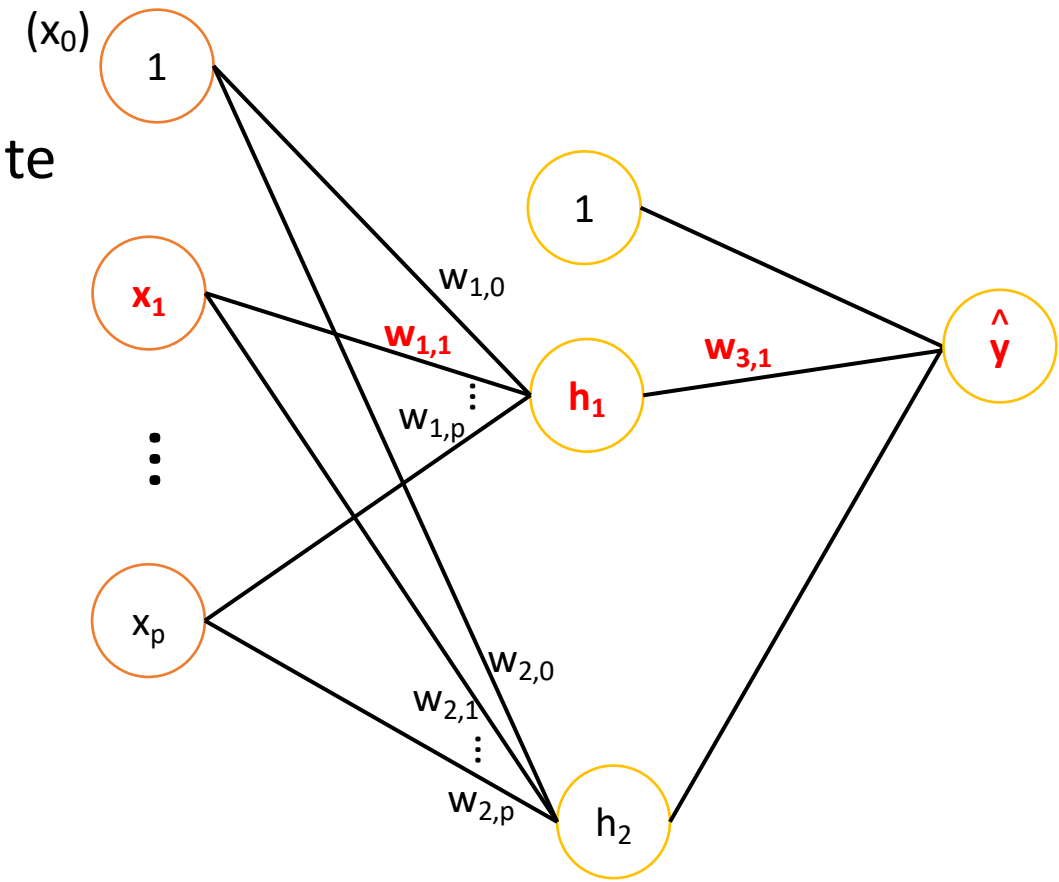
$$h_1 = \sigma(s_1)$$

$$s_1 = \sum_{i=0}^{p} w_{1,i} x_i$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Aside: Backprop in the brain

- The update for $w_{1,1}$ depends on
  - $y - \hat{y}$ (error)
  - h_1 The hidden element $w_{1,1}$ is used to compute
  - x_1 the input $w_{1,1}$ is multiplied with
  - $w_{3,1}$
- What does that mean about the brain?
  - To alter the synapse between x_1 and h_1



Resources

- Programming resources for training your own NNs
  - Tensorflow https://www.tensorflow.org/
  - Keras https://keras.io/
  - Pytorch https://pytorch.org/
  - For intuition: http://playground.tensorflow.org/
- Short course on deep learning (Nando De Freitas)
  - https://www.youtube.com/playlist?list=PLjK8ddCbDMphIMSXn-w1IjyYpHU3DaUYw
- Commentary on AlphaGo
  - https://www.youtube.com/watch?v=UMm0XaCFTJQ
  - https://www.youtube.com/watch?v=g-dKXOlsf98
- Other fun videos
  - Geoff Hinton is in this one! Neural Net stuff is towards the end
    - https://www.youtube.com/watch?v=yxxRAHVtafI
  - Fei Fei Li's Ted Talk (Creator of ImageNet)
    - https://www.ted.com/talks/fei_fei_li_how_we_re_teaching_computers_to_understand_pictures?language=en