Warm-up as You Log In

Suppose we have a function that takes in a vector and squares each element individually, returning another vector, $\boldsymbol{y} = f(\boldsymbol{x})$.

$$f\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) \rightarrow \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \end{bmatrix} \qquad f\left(\begin{bmatrix} 7 \\ 3 \\ 5 \end{bmatrix}\right) \rightarrow \begin{bmatrix} 49 \\ 9 \\ 25 \end{bmatrix}$$

What is $\partial\boldsymbol{y}/\partial\boldsymbol{x}$?

Announcements

Assignments

- HW5
  - Due Mon, 10/26, 11:59 pm
  - Start early

Recitation

- No recitation this Friday

Introduction to Machine Learning

Neural Networks

Instructor: Pat Virtue

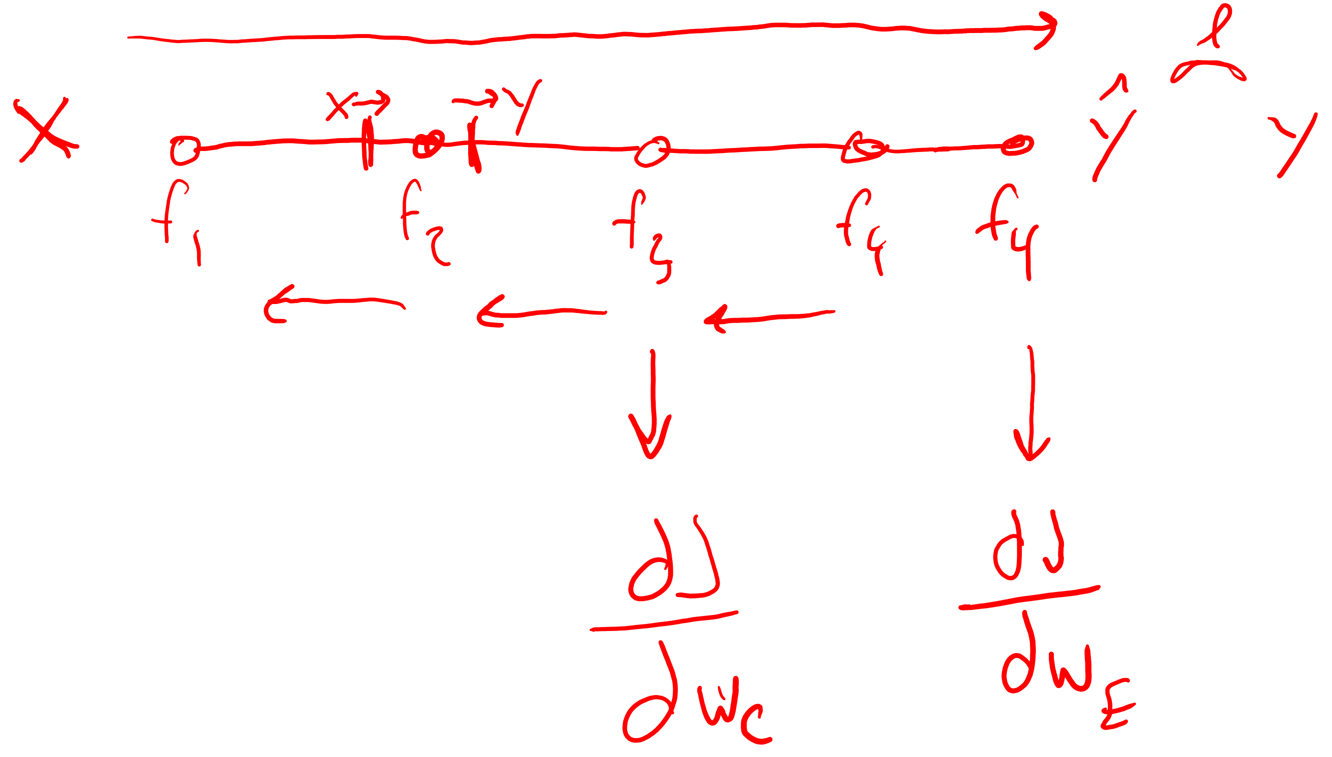
Plan

Last Time

- Neural Networks
  - Building blocks
  - Optimization

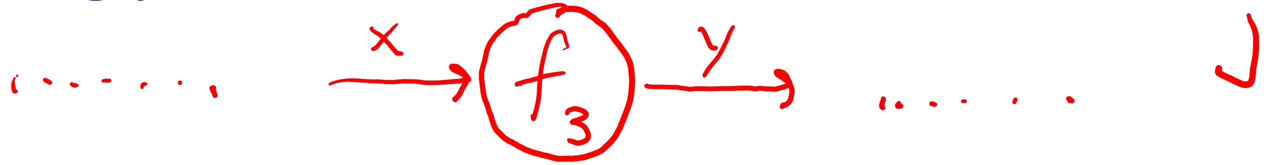
Today

- Neural Networks
  - Wrap-up calculus
  - Universal Approximation Theorem
  - Convolutional neural networks



Backpropagation (so-far)

Compute derivatives per layer, utilizing previous derivatives

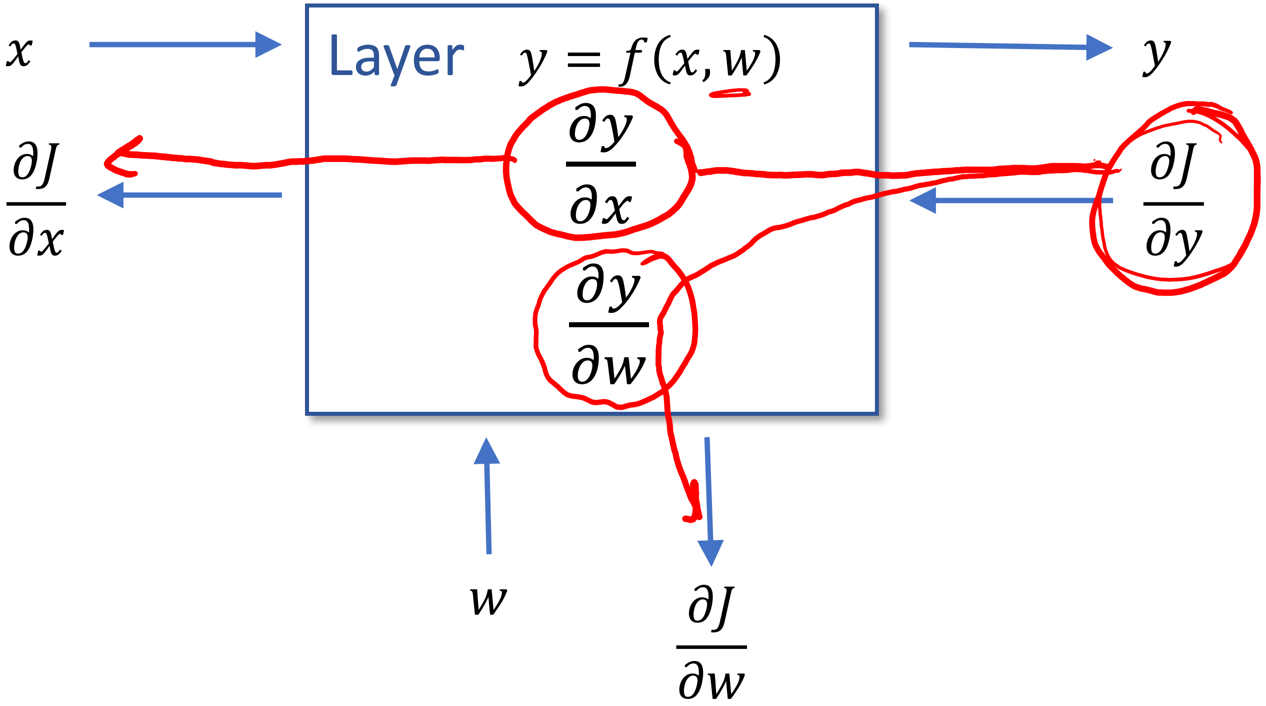


Objective: $J(\boldsymbol{w})$

Arbitrary layer: $y = f(x, w)$

Need:

- $\dfrac{\partial J}{\partial x} = \dfrac{\partial J}{\partial y} \dfrac{\partial y}{\partial x}$
- $\dfrac{\partial J}{\partial w} = \dfrac{\partial J}{\partial y} \dfrac{\partial y}{\partial w}$



Matrix Calculus

- Jacobian: Vector in, vector out
- Numerator-layout   num out × num in

$$\boldsymbol{y} = f(\boldsymbol{x}) \qquad \boldsymbol{y} \in \mathbb{R}^N, \quad \boldsymbol{x} \in \mathbb{R}^M, \quad \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} \in \mathbb{R}^{N \times M}$$

M

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_N}{\partial x_1} & \cdots & \frac{\partial y_N}{\partial x_M} \end{bmatrix}$$

Matrix Calculus

Vector in, scalar out

Numerator-layout

$y = f(\boldsymbol{x}) \qquad y \in \mathbb{R}, \quad \boldsymbol{x} \in \mathbb{R}^M, \quad \frac{\partial y}{\partial \boldsymbol{x}} \in \mathbb{R}^{1 \times M}$

$$\frac{\partial y}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \cdots & \frac{\partial y}{\partial x_M} \end{bmatrix}$$

Matrix Calculus

Scalar in, vector out

Numerator-layout

$\boldsymbol{y} = f(x) \qquad \boldsymbol{y} \in \mathbb{R}^N, \quad x \in \mathbb{R}, \quad \frac{\partial \boldsymbol{y}}{\partial x} \in \mathbb{R}^{N\times 1}$

$$\frac{\partial \boldsymbol{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \vdots \\ \frac{\partial y_N}{\partial x} \end{bmatrix}$$

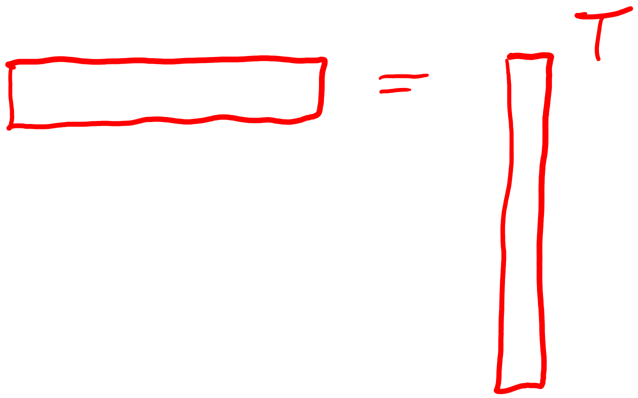
Matrix Calculus

Gradient: Vector in, scalar out

Transpose of numerator-layout

$y = f(\boldsymbol{x}) \qquad y \in \mathbb{R}, \quad \boldsymbol{x} \in \mathbb{R}^M, \quad \frac{\partial y}{\partial \boldsymbol{x}} \in \mathbb{R}^{1\times M}, \quad \nabla_{\boldsymbol{x}} f \in \mathbb{R}^{M\times 1}$

$$\frac{\partial y}{\partial \boldsymbol{x}} = (\nabla_{\boldsymbol{x}} f)^T$$



Matrix Calculus

$W \vec{x} \rightarrow \vec{a}$

Matrix in, scalar out

Keep same dimensions as matrix

$y = f(\boldsymbol{X}) \quad y \in \mathbb{R}, \quad \boldsymbol{X} \in \mathbb{R}^{N \times M}, \quad \frac{\partial y}{\partial \boldsymbol{X}} \in \mathbb{R}^{N \times M}$

$$\frac{\partial y}{\partial \boldsymbol{X}} = \begin{bmatrix} \frac{\partial y}{\partial X_{1,1}} & \cdots & \frac{\partial y}{\partial X_{1,M}} \\ \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial X_{N,1}} & \cdots & \frac{\partial y}{\partial X_{N,M}} \end{bmatrix}$$

Warm-up as You Log In

$2\vec{x}$ ←

Suppose we have a function that takes in a vector and squares each element individually, returning another vector, $\boldsymbol{y} = f(\boldsymbol{x})$.

$$f\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) \rightarrow \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \end{bmatrix} \qquad f\left(\begin{bmatrix} 7 \\ 3 \\ 5 \end{bmatrix}\right) \rightarrow \begin{bmatrix} 49 \\ 9 \\ 25 \end{bmatrix}$$

What is $\partial \boldsymbol{y} / \partial \boldsymbol{x}$?

$$= \begin{bmatrix} 2x_1 & 0 & 0 \\ 0 & 2x_2 & 0 \\ 0 & 0 & 2x_3 \end{bmatrix}$$

$$\frac{dy_i}{dx_j} = 2x_j \, \mathbb{1}(i=j)$$

Calculus Chain Rule

Scalar:

$$y = f(z)$$

$$z = g(x)$$

$$\frac{dy}{dx} = \frac{dy}{dz}\frac{dz}{dx}$$

Multivariate:

$$y = f(\mathbf{z})$$

$$\mathbf{z} = g(x)$$

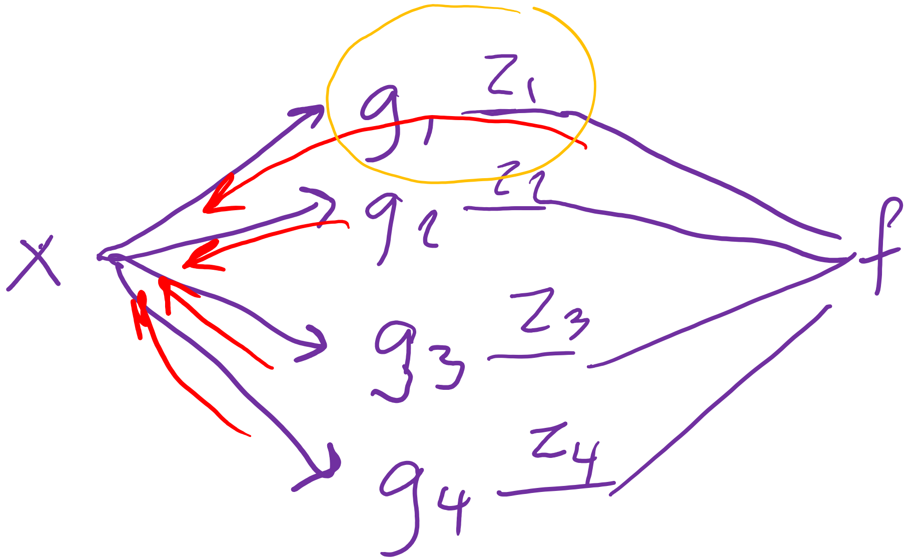
$$\frac{dy}{dx} = \sum_j \frac{\partial y}{\partial z_j}\frac{\partial z_j}{\partial x}$$

Multivariate:

$$\mathbf{y} = f(\mathbf{z})$$

$$\mathbf{z} = g(\mathbf{x})$$

$$\frac{dy_i}{dx_k} = \sum_j \frac{\partial y_i}{\partial z_j}\frac{\partial z_j}{\partial x_k}$$



Piazza Poll 1

out-by-in

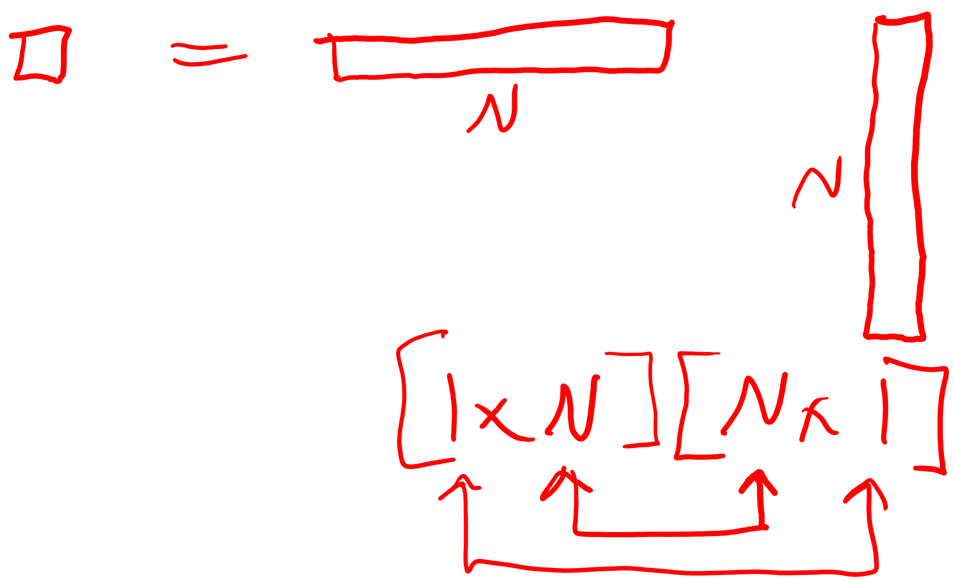
$y = f(\boldsymbol{z}) \qquad y \in \mathbb{R},\ \boldsymbol{z} \in \mathbb{R}^N,\ x \in \mathbb{R}$

$\boldsymbol{z} = g(x)$

Select all that apply

$\frac{\partial y}{\partial x} = \cdots$

$\frac{\partial y}{\partial x} \qquad \frac{\partial y}{\partial \vec{z}} \qquad \frac{\partial \vec{z}}{\partial x}$



75% A. $\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial x}$

B. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \frac{\partial \boldsymbol{z}}{\partial x}$

C. $\frac{\partial y}{\partial \boldsymbol{z}} \left(\frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

D. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \left(\frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

50% E. $\left(\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

F. None of the above

Piazza Poll 1

$y = f(\boldsymbol{z}) \qquad y \in \mathbb{R},\ \boldsymbol{z} \in \mathbb{R}^N,\ x \in \mathbb{R}$

$\boldsymbol{z} = g(x)$

Select all that apply

$\frac{\partial y}{\partial x} = \cdots$

A. $\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial x}$

B. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \frac{\partial \boldsymbol{z}}{\partial x}$

C. $\frac{\partial y}{\partial \boldsymbol{z}} \left(\frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

D. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \left(\frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

E. $\left(\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial x}\right)^T$

F. None of the above

out x in

Piazza Poll 2

$y = f(\boldsymbol{z}) \qquad y \in \mathbb{R},\ \boldsymbol{z} \in \mathbb{R}^N,\ \boldsymbol{x} \in \mathbb{R}^M$

$\boldsymbol{z} = g(\boldsymbol{x})$

Select all that apply

$\frac{\partial y}{\partial \boldsymbol{x}} = \cdots$

A. $\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$

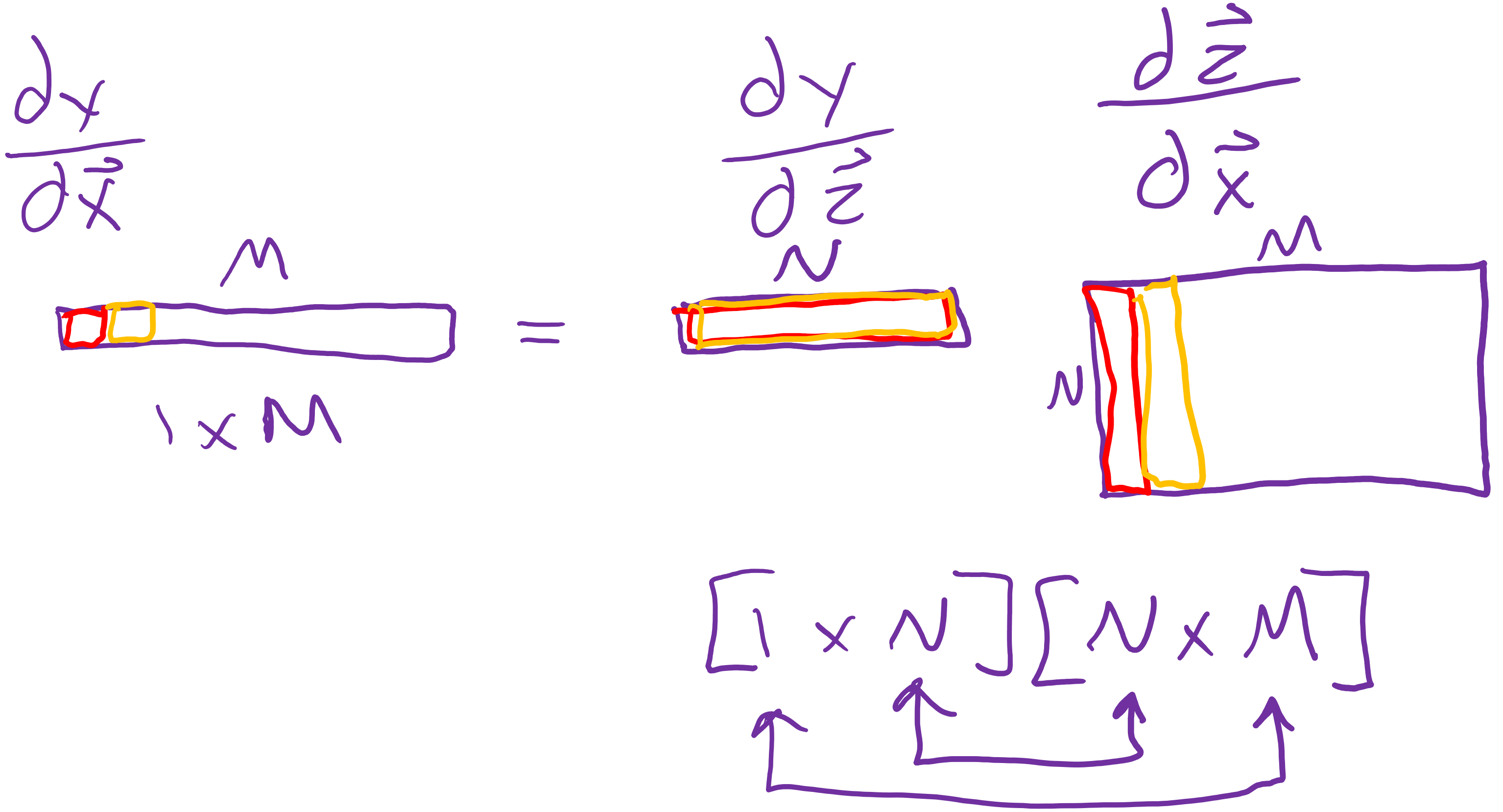
B. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$

C. $\frac{\partial y}{\partial \boldsymbol{z}} \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

D. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

E. $\left(\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

F. None of the above



Piazza Poll 2

$y = f(\boldsymbol{z})$

$\boldsymbol{z} = g(\boldsymbol{x})$

Select all that apply

$\frac{\partial y}{\partial \boldsymbol{x}} = \cdots$

A. $\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$

B. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$

C. $\frac{\partial y}{\partial \boldsymbol{z}} \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

D. $\left(\frac{\partial y}{\partial \boldsymbol{z}}\right)^T \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

E. $\left(\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)^T$

F. None of the above

Network Optimization

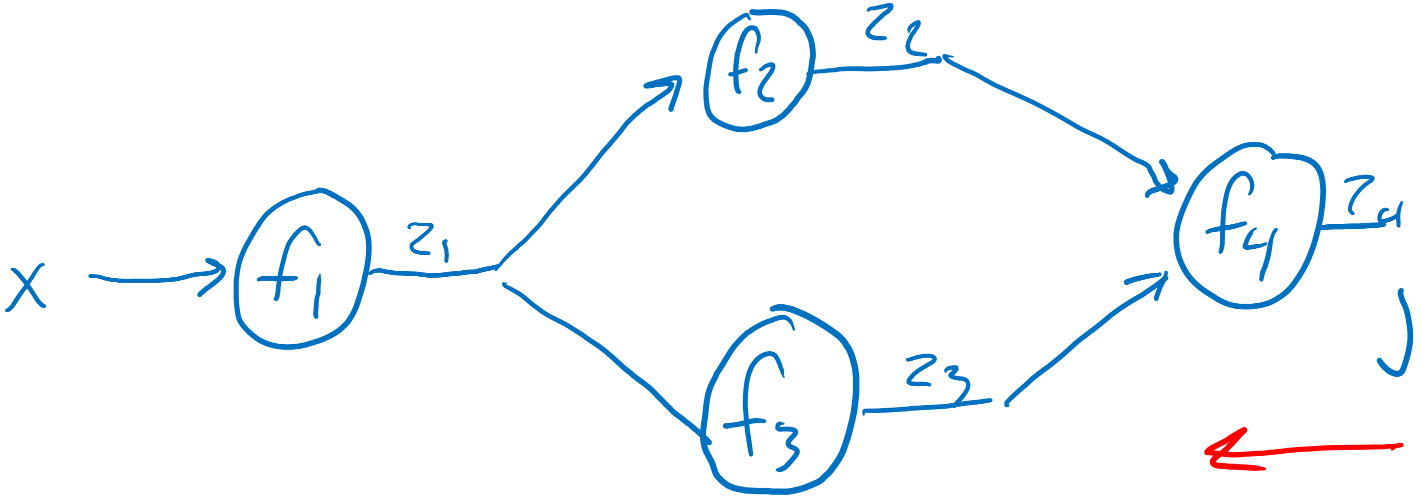
$$J(\boldsymbol{w}) = z_4$$

$$z_4 = f_4(w_D, w_E, z_2, z_3)$$

$$z_3 = f_3(w_C, z_1)$$

$$z_2 = f_2(w_B, z_1)$$

$$z_1 = f_1(w_A, x)$$



Need multivariate chain rule!

$$\frac{\partial J}{\partial w_E}, \frac{\partial J}{\partial w_D}, \frac{\partial J}{\partial w_C}, \frac{\partial J}{\partial w_B}, \frac{\partial J}{\partial w_A}$$

Network Optimization

$J(\boldsymbol{w}) = z_4$

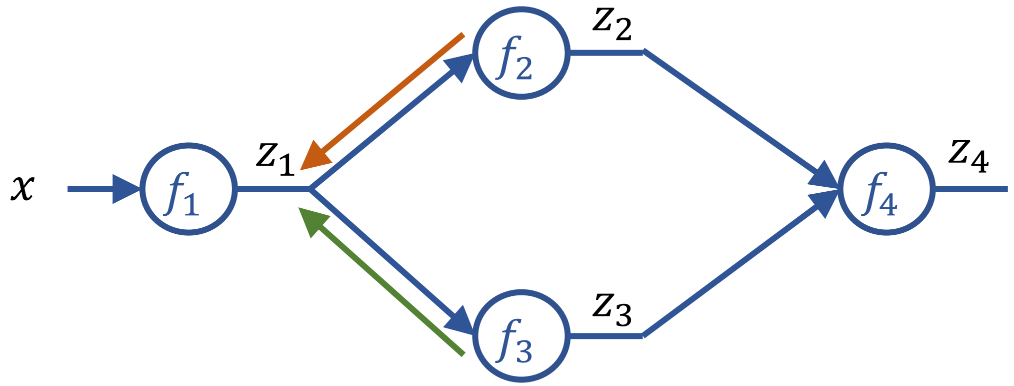
$z_4 = f_4(w_D, w_E, z_2, z_3)$

$z_3 = f_3(w_C, z_1)$

$z_2 = f_2(w_B, z_1)$

$z_1 = f_1(w_A, x)$

Need multivariate chain rule!



$$\frac{\partial J}{\partial w_E} = \frac{\partial J}{\partial z_4}\frac{\partial z_4}{\partial w_E}$$

$$\frac{\partial J}{\partial w_D} = \frac{\partial J}{\partial z_4}\frac{\partial z_4}{\partial w_D}$$

$$\frac{\partial J}{\partial z_3} = \frac{\partial J}{\partial z_4}\frac{\partial z_4}{\partial z_3}$$

$$\frac{\partial J}{\partial z_2} = \frac{\partial J}{\partial z_4}\frac{\partial z_4}{\partial z_2}$$

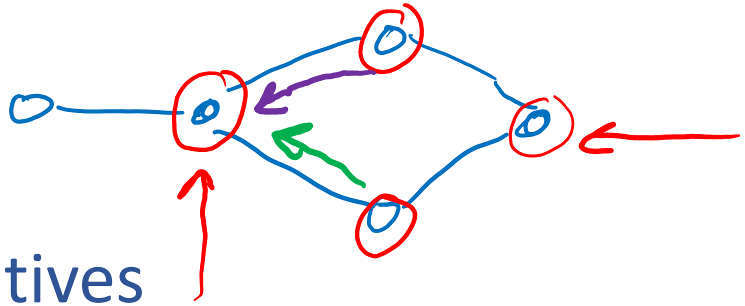
$$\frac{\partial J}{\partial w_C} = \frac{\partial J}{\partial z_3}\frac{\partial z_3}{\partial w_C}$$

$$\frac{\partial J}{\partial w_B} = \frac{\partial J}{\partial z_2}\frac{\partial z_2}{\partial w_B}$$

$$\frac{\partial J}{\partial z_1} = \frac{\partial J}{\partial z_2}\frac{\partial z_2}{\partial z_1} + \frac{\partial J}{\partial z_3}\frac{\partial z_3}{\partial z_1}$$

$$\frac{\partial J}{\partial w_A} = \frac{\partial J}{\partial z_1}\frac{\partial z_1}{\partial w_A}$$

Backpropagation (updated)



Compute derivatives per layer, utilizing previous derivatives

Objective: $J(\boldsymbol{w})$

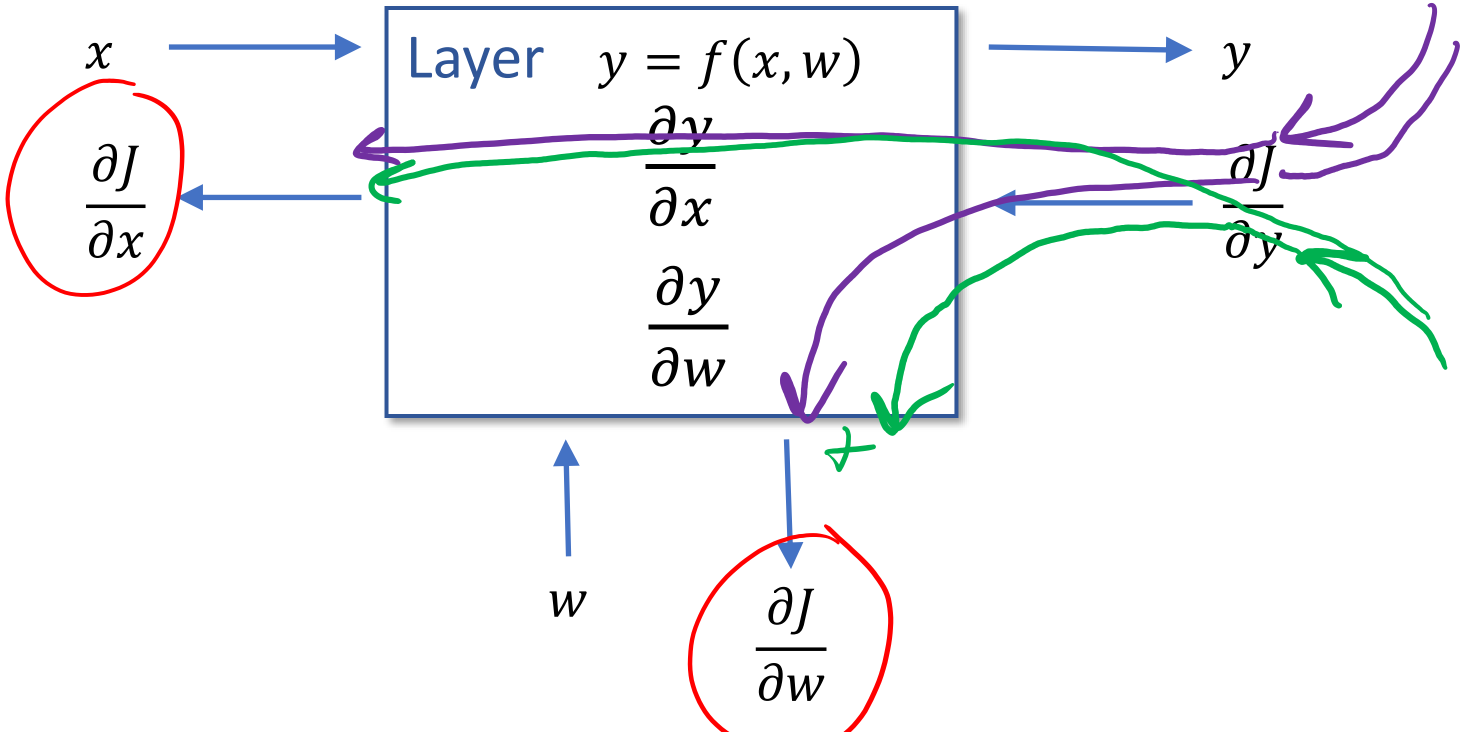
Arbitrary layer: $y = f(x, w)$

Init:

- $\frac{\partial J}{\partial x} = 0$
- $\frac{\partial J}{\partial w} = 0$

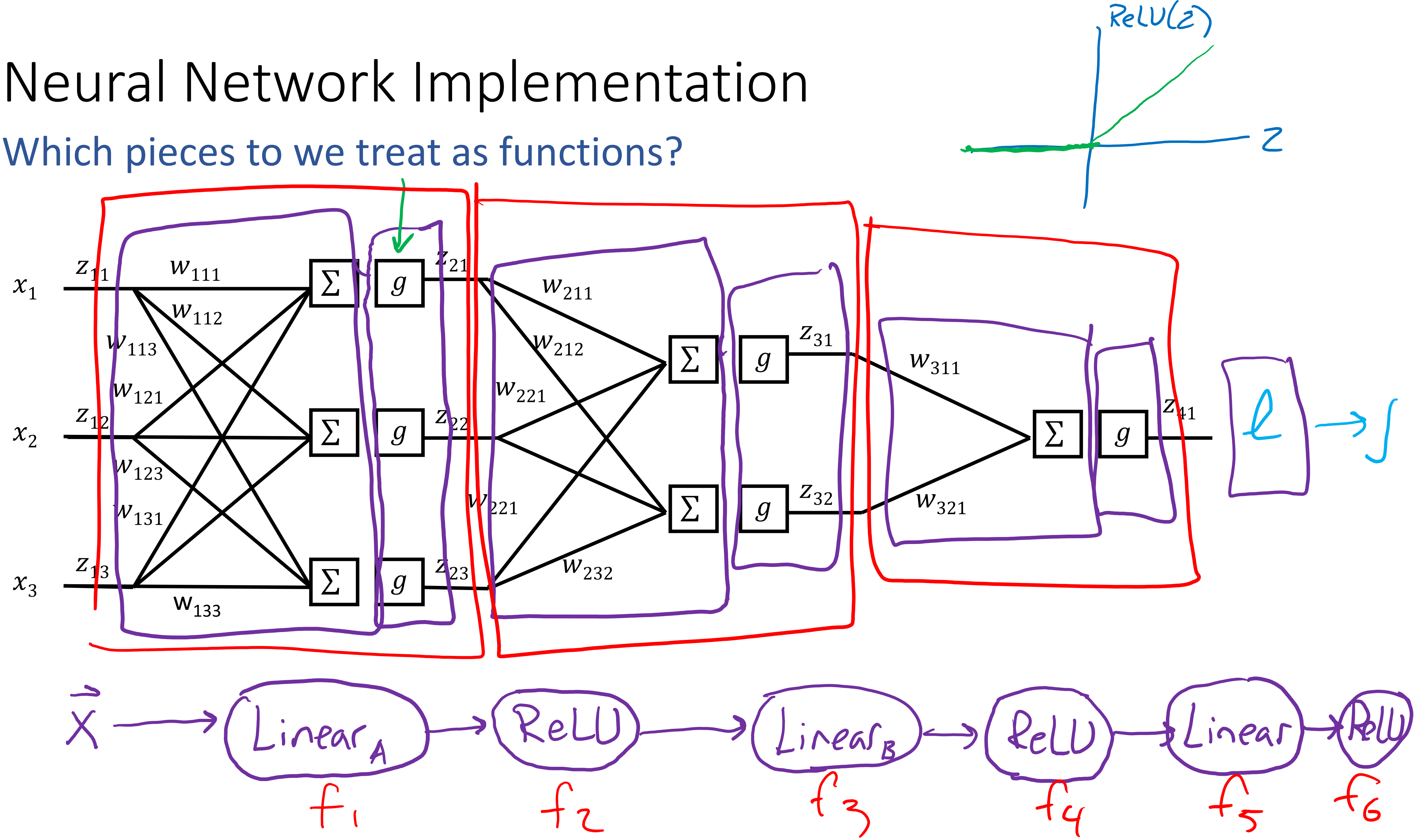
Compute:

- $\frac{\partial J}{\partial x} += \frac{\partial J}{\partial y}\frac{\partial y}{\partial x}$
- $\frac{\partial J}{\partial w} += \frac{\partial J}{\partial y}\frac{\partial y}{\partial w}$



Neural Network Implementation

Which pieces to we treat as functions?



Neural Networks Properties

$\vec{x} \rightarrow \vec{z}_1 \rightarrow \vec{z}_2$

Practical considerations

- Large number of neurons
  - Danger for overfitting
- Modelling assumptions vs data assumptions trade-off
- Gradient descent can easily get stuck local optima

What if there are no non-linear activations?

- A deep neural network with only linear layers can be reduced to an exactly equivalent single linear layer

Universal Approximation Theorem:

- A two-layer neural network with a sufficient number of neurons can approximate any continuous function to any desired accuracy.

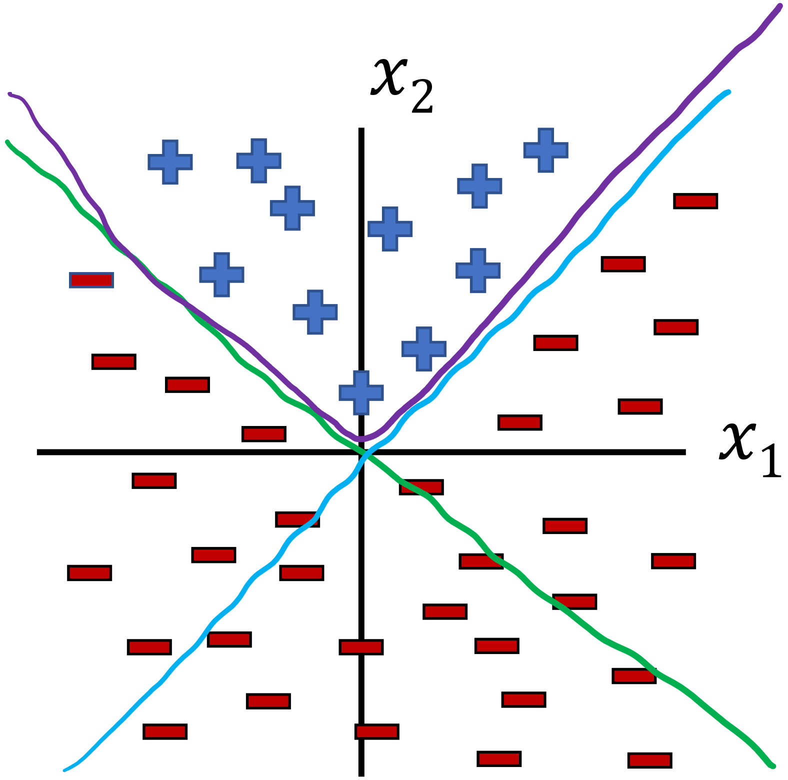
Classification Design Challenge

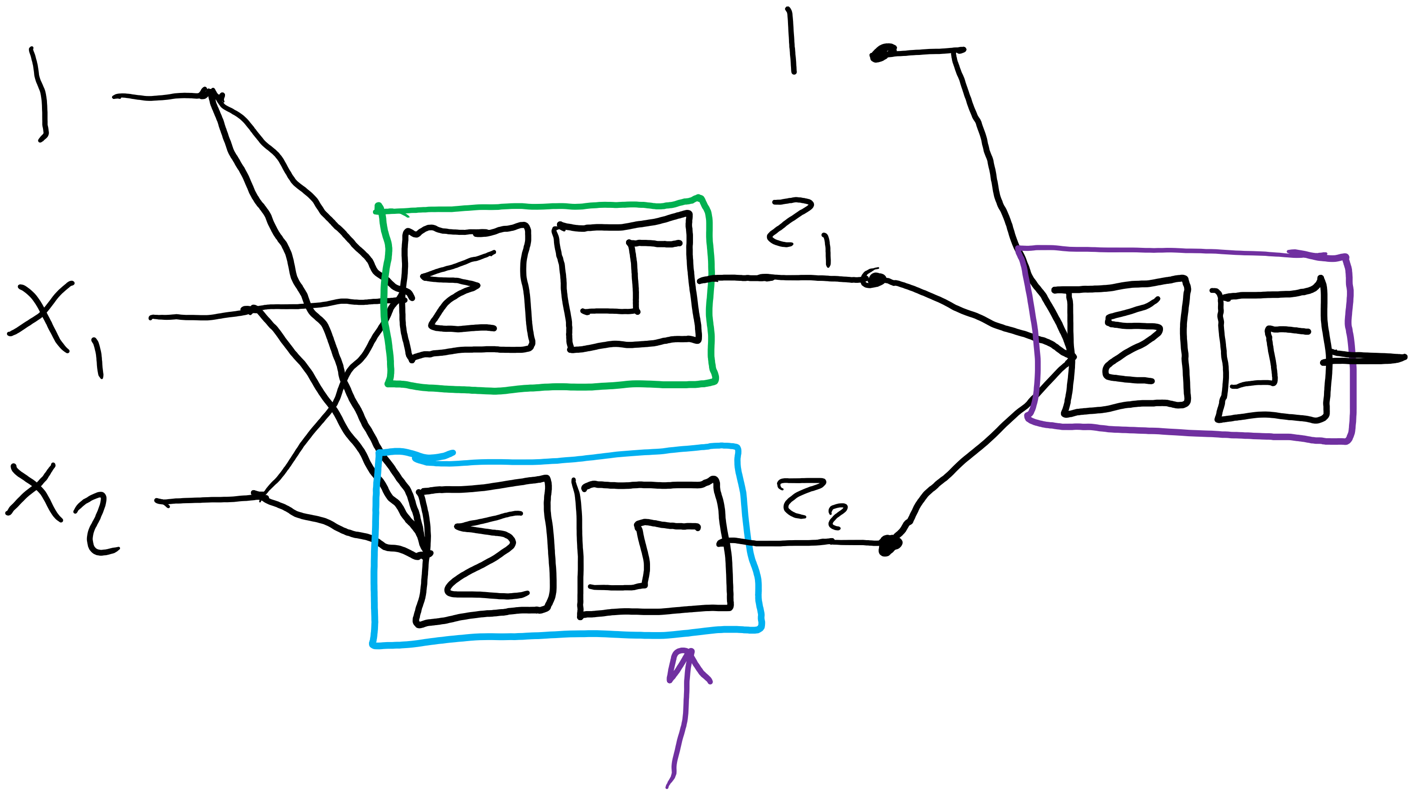
How could you configure three specific perceptrons to classify this data?

$$h_A(\boldsymbol{x}) = sign(\boldsymbol{w}_A^T\boldsymbol{x} + b_A)$$

$$h_B(\boldsymbol{x}) = sign(\boldsymbol{w}_B^T\boldsymbol{x} + b_B)$$

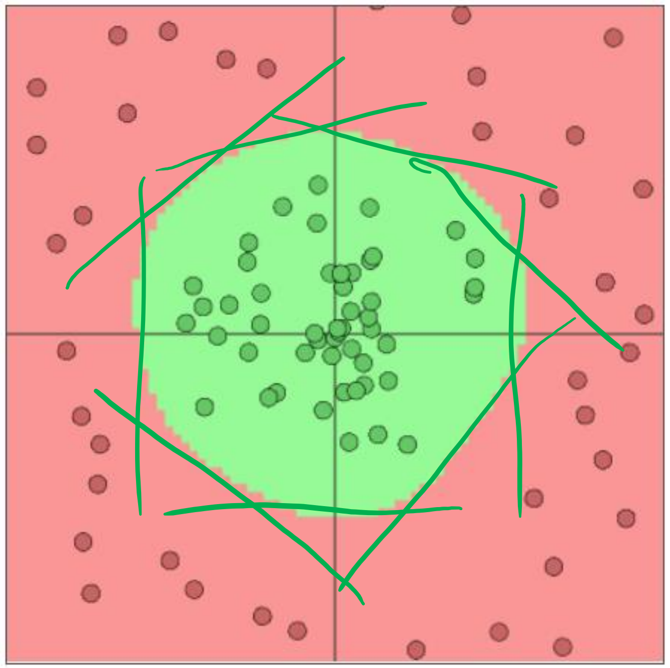
$$h_C(\underset{z}{\cancel{\boldsymbol{x}}}) = sign(\boldsymbol{w}_C^T\underset{z}{\cancel{\boldsymbol{x}}} + b_C)$$





Network to Approximate Binary Classification

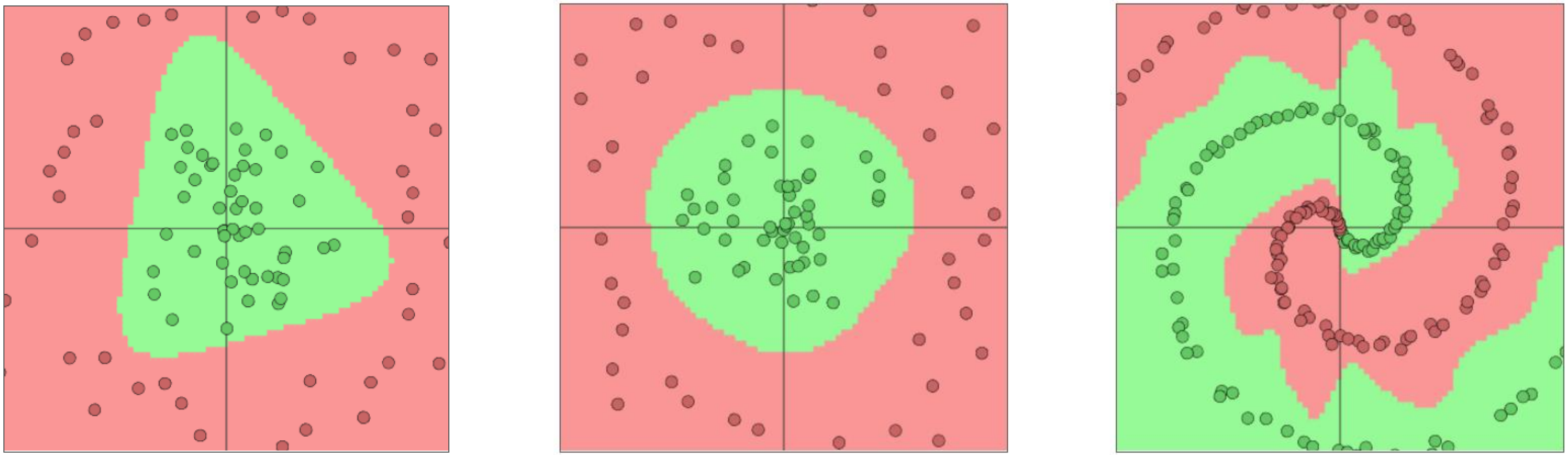
Approximate arbitrary decision boundary



https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Network to Approximate Binary Classification

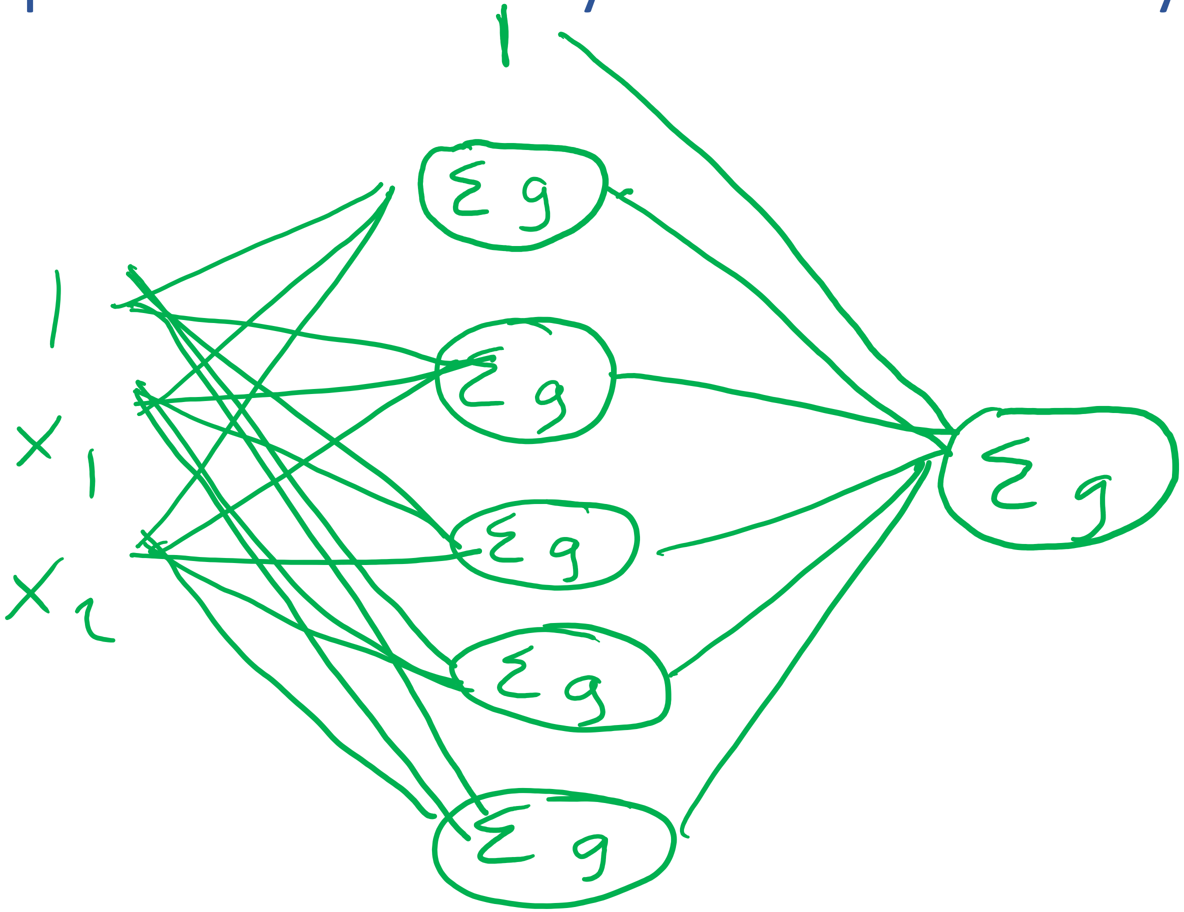
Approximate arbitrary decision boundary

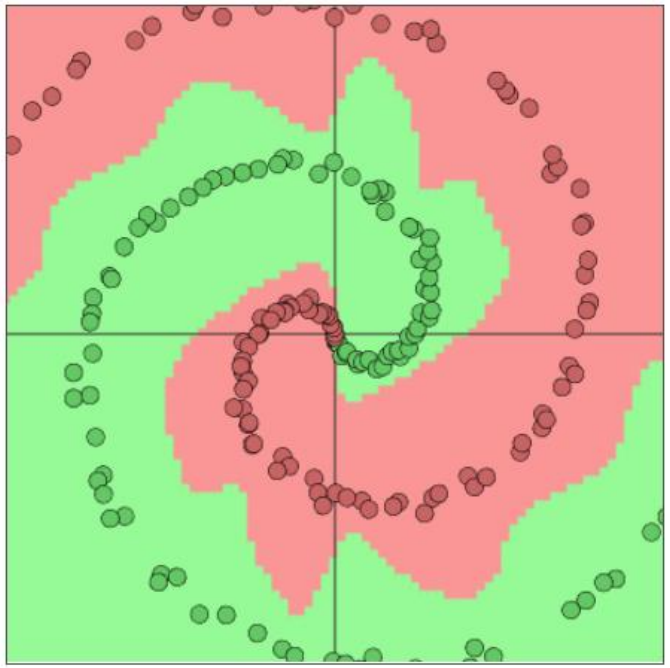


https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Network to Approximate Binary Classification

Approximate arbitrary decision boundary

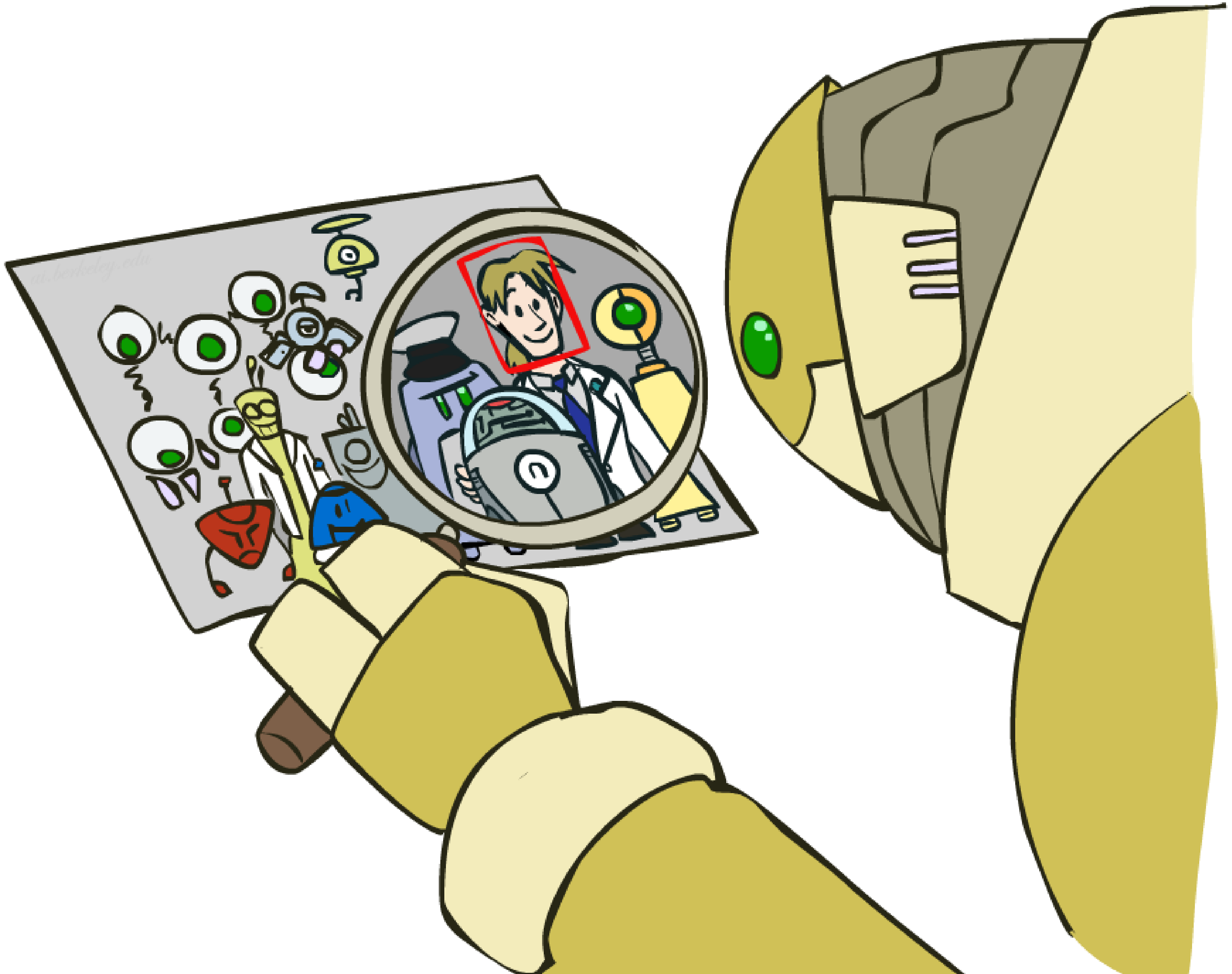




https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Convolutional Neural Nets

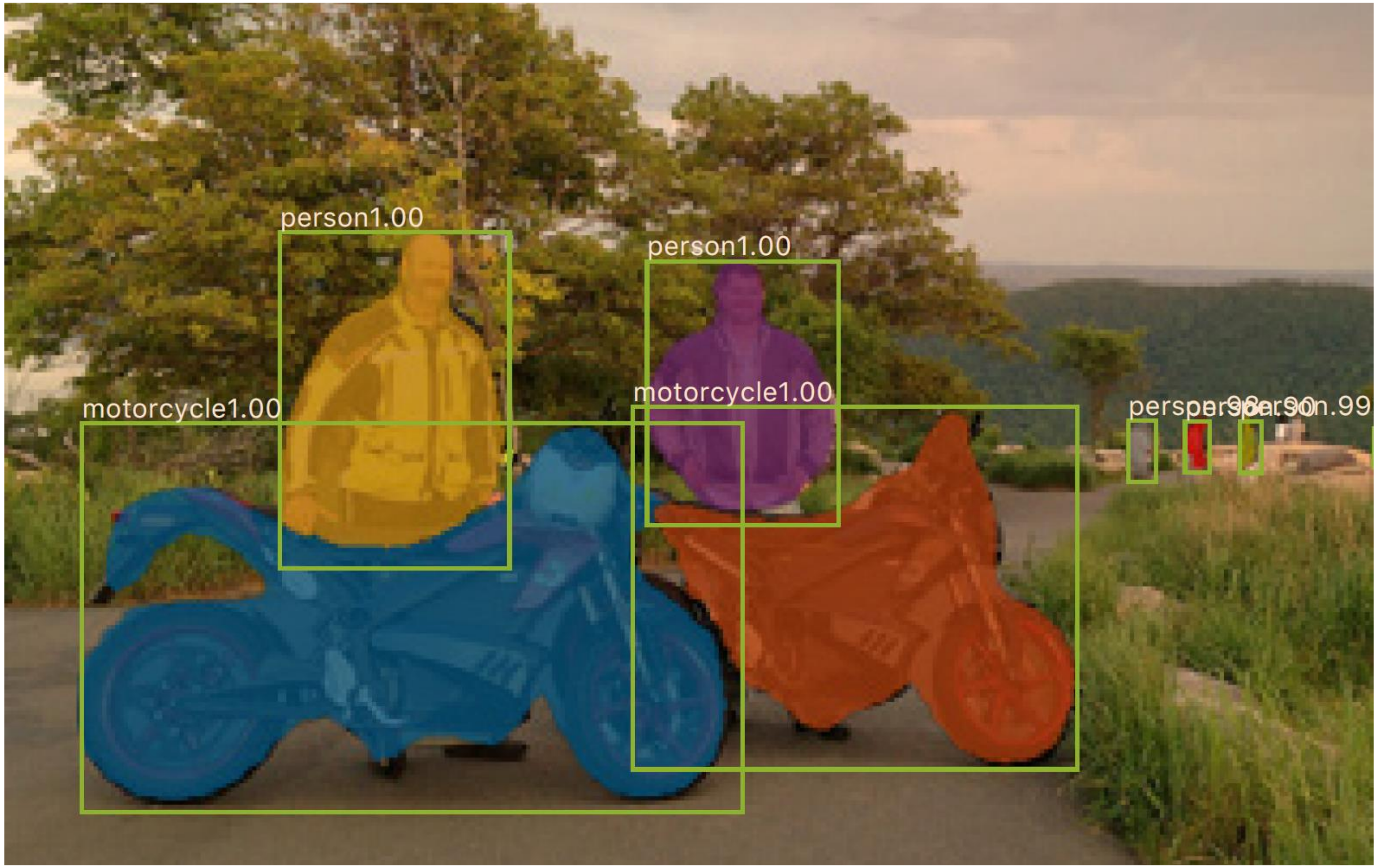
Computer Vision: How far along are we?



Computer Vision: How far along are we?

Terminator 2, 1991 https://www.youtube.com/watch?v=9MeaaCwBW28

Computer Vision: How far along are we?



0.2 seconds per image

Mask R-CNN

He, Kaiming, et al. "Mask R-CNN." *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017.

Computer Vision: How far along are we?

"My CPU is a neural net processor, a learning computer"

Terminator 2, 1991

Computer Vision: Autonomous Driving

Tesla, Inc: https://vimeo.com/192179726

Computer Vision: Domain Transfer

CycleGAN

Jun-Yan Zhu*, Taesung Park*, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", ICCV 2017.

Outline

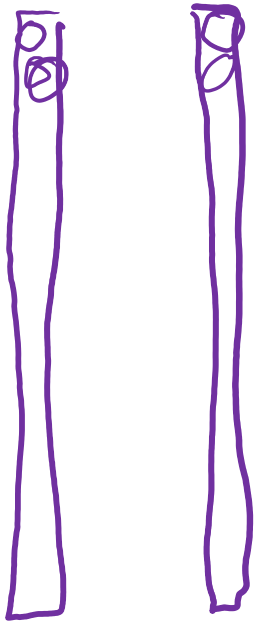
1. Measuring the current state of computer vision
2. Why convolutional neural networks
   - Old school computer vision
   - Image features and classification
3. Convolution “nuts and bolts”

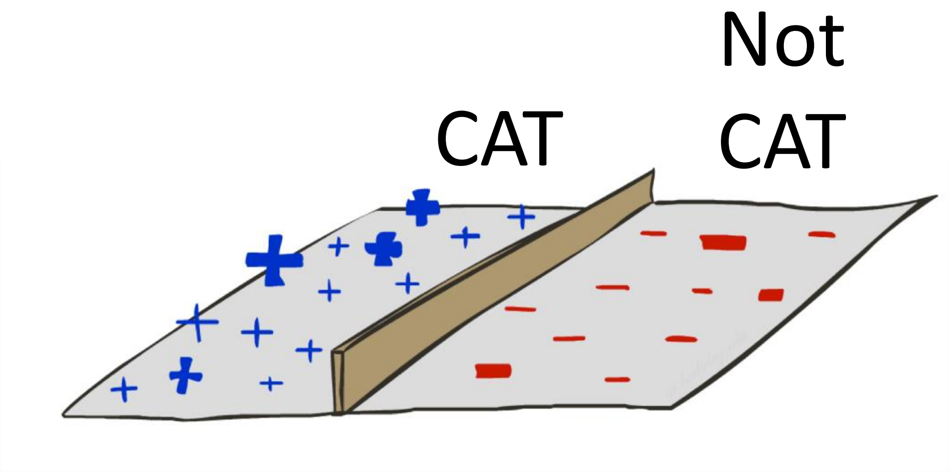
Image Classification

What's the problem with just directly classifying raw pixels in high dimensional space?

x m

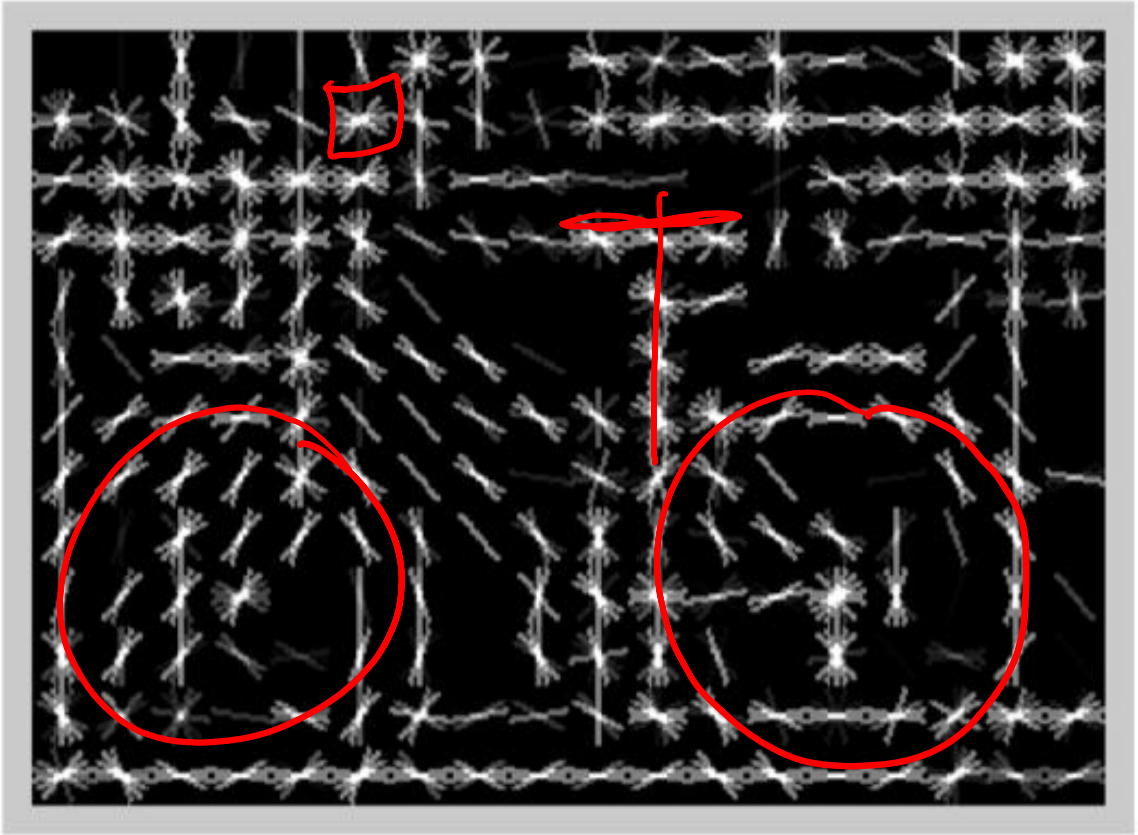
$\vec{x}$ $\vec{w}$





CAT

Image Classification

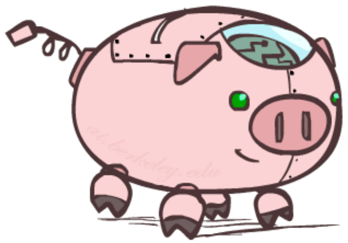


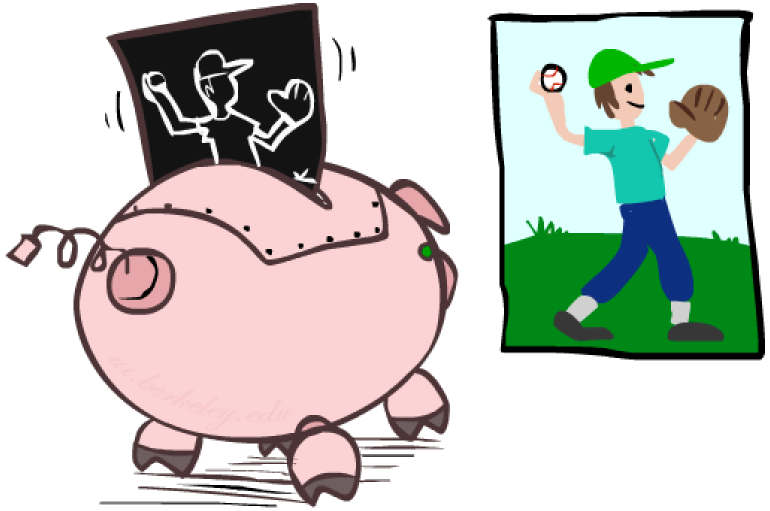
$$X \xrightarrow{\text{edge}} Z_1 \xrightarrow{\text{HOG}} Z_2 \xrightarrow{\text{Part}} Z_3 \rightarrow \hat{Y}$$

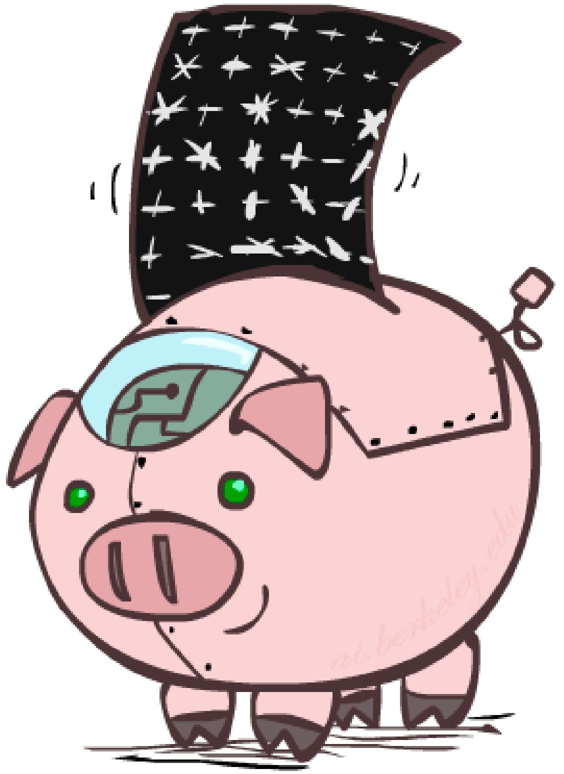
[Dalal and Triggs, 2005]

HoG Filter

HoG: Histogram of oriented gradients



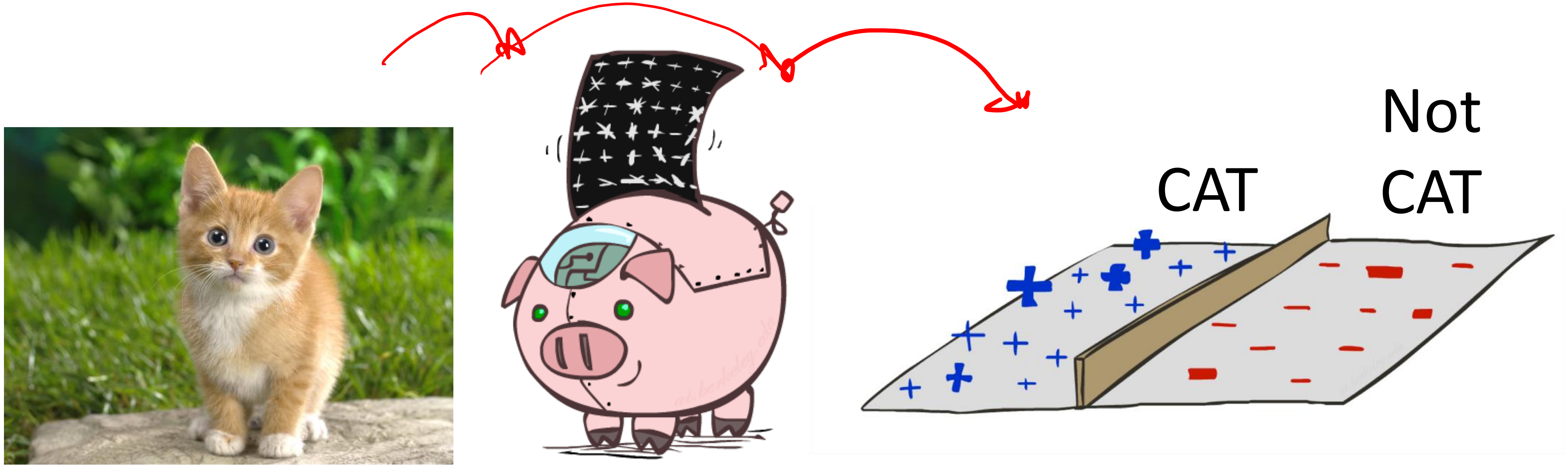




[Dalal and Triggs, 2005]

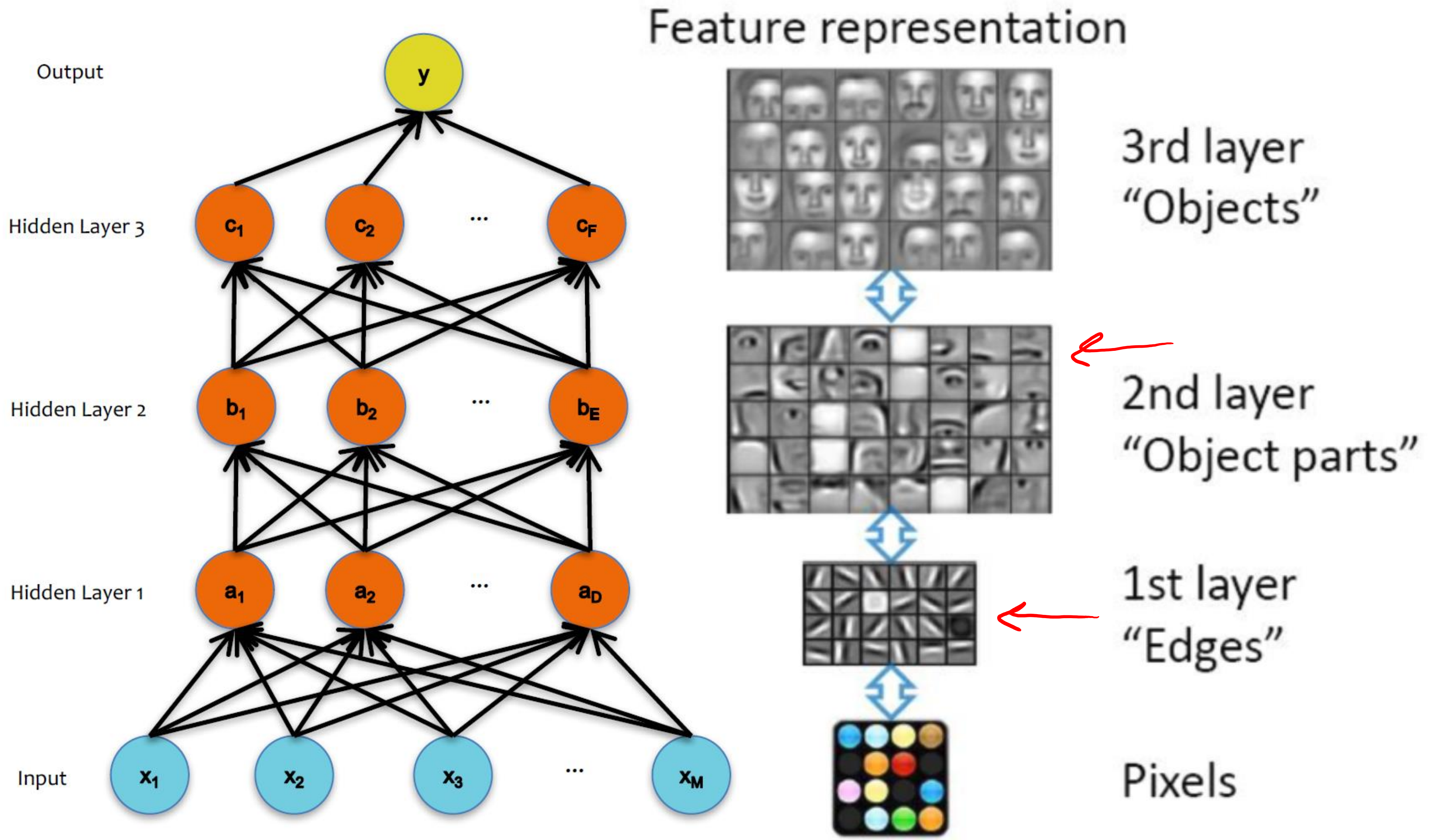
Image Classification

HOG features passed to a linear classifier (logistic regression / SVM)



CAT

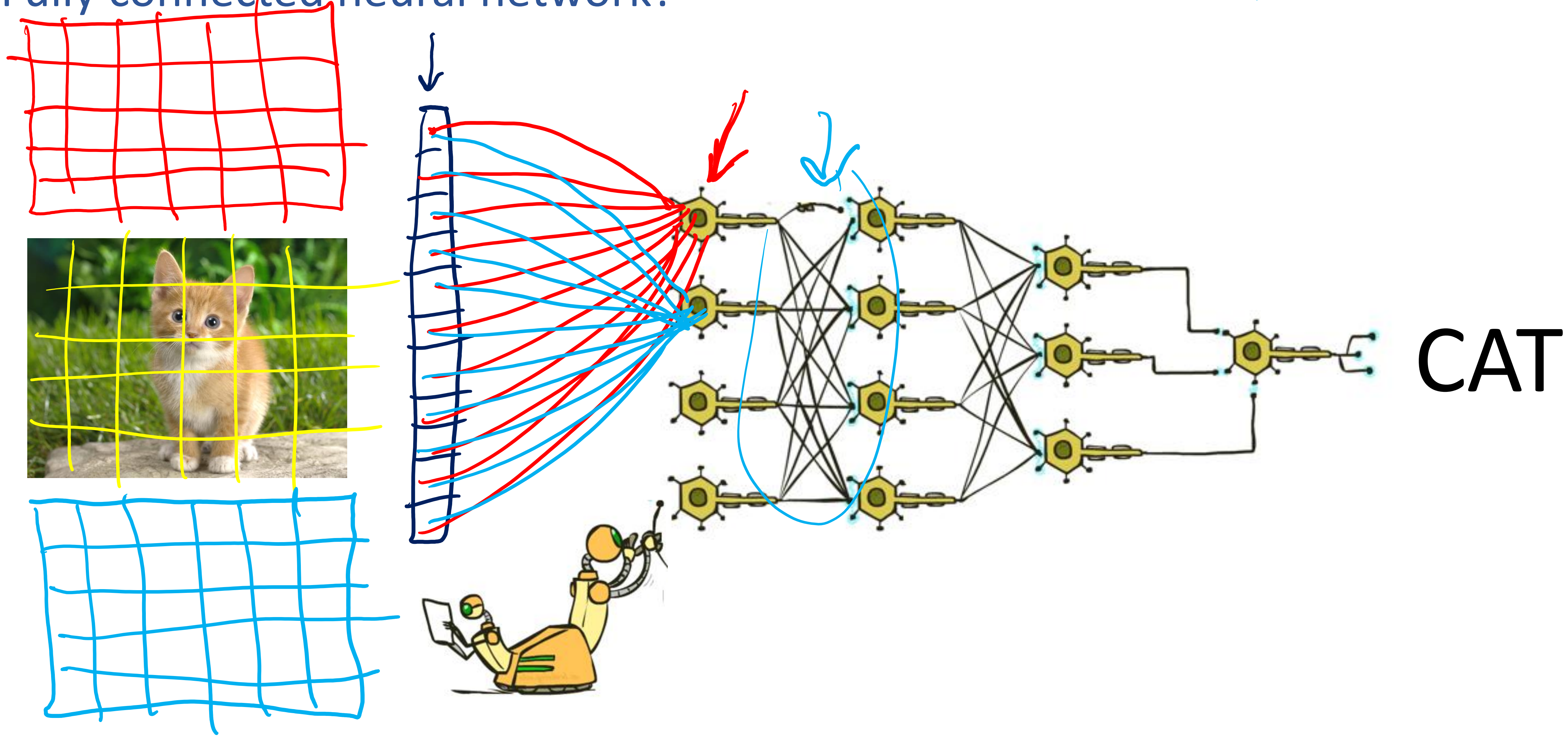
Classification: Learning Features

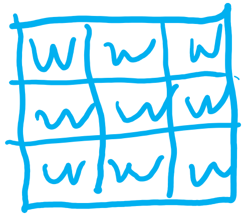


Example from Honglak Lee (NIPS 2010)

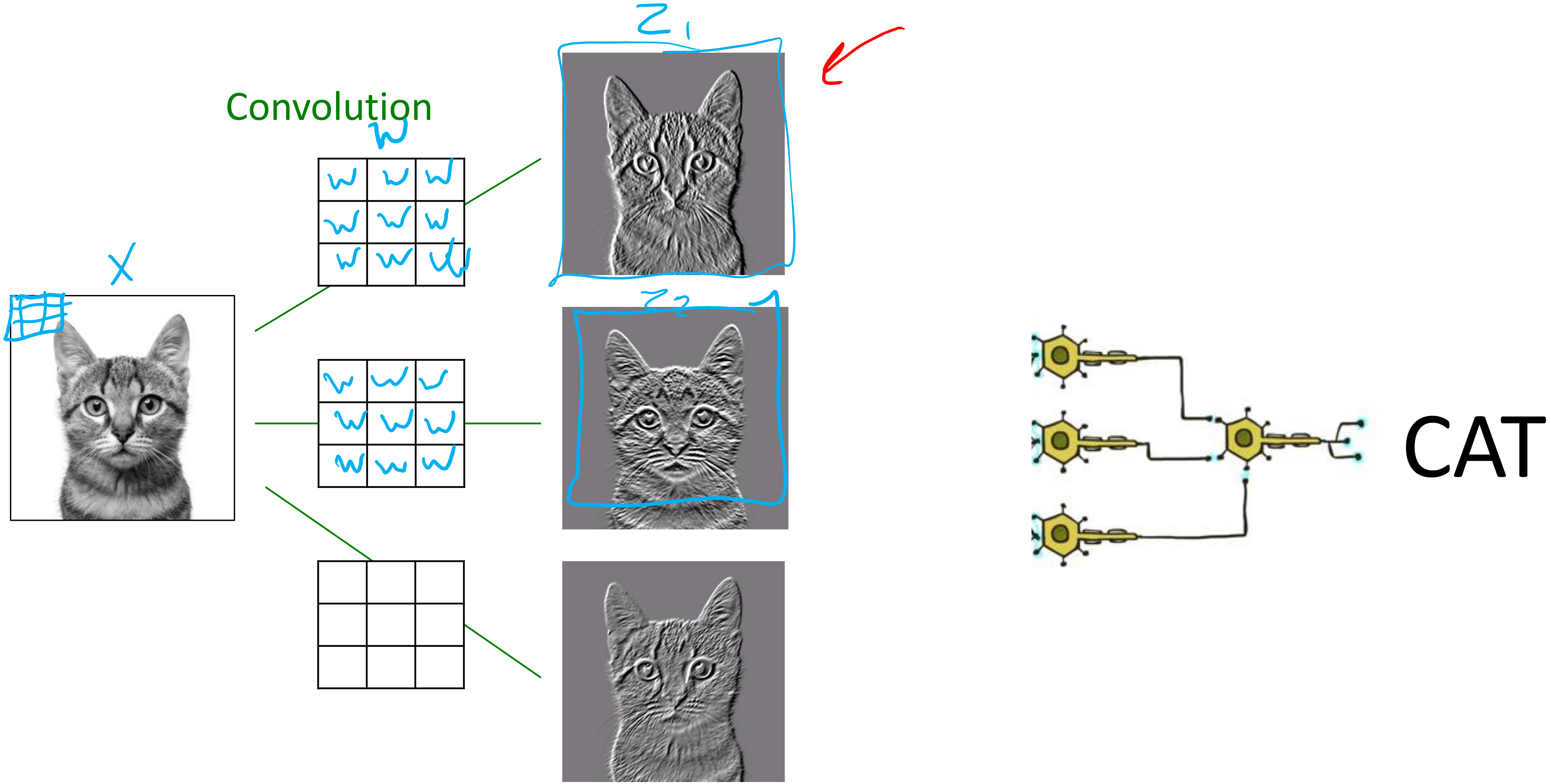
Classification: Deep Learning

Fully connected neural network?



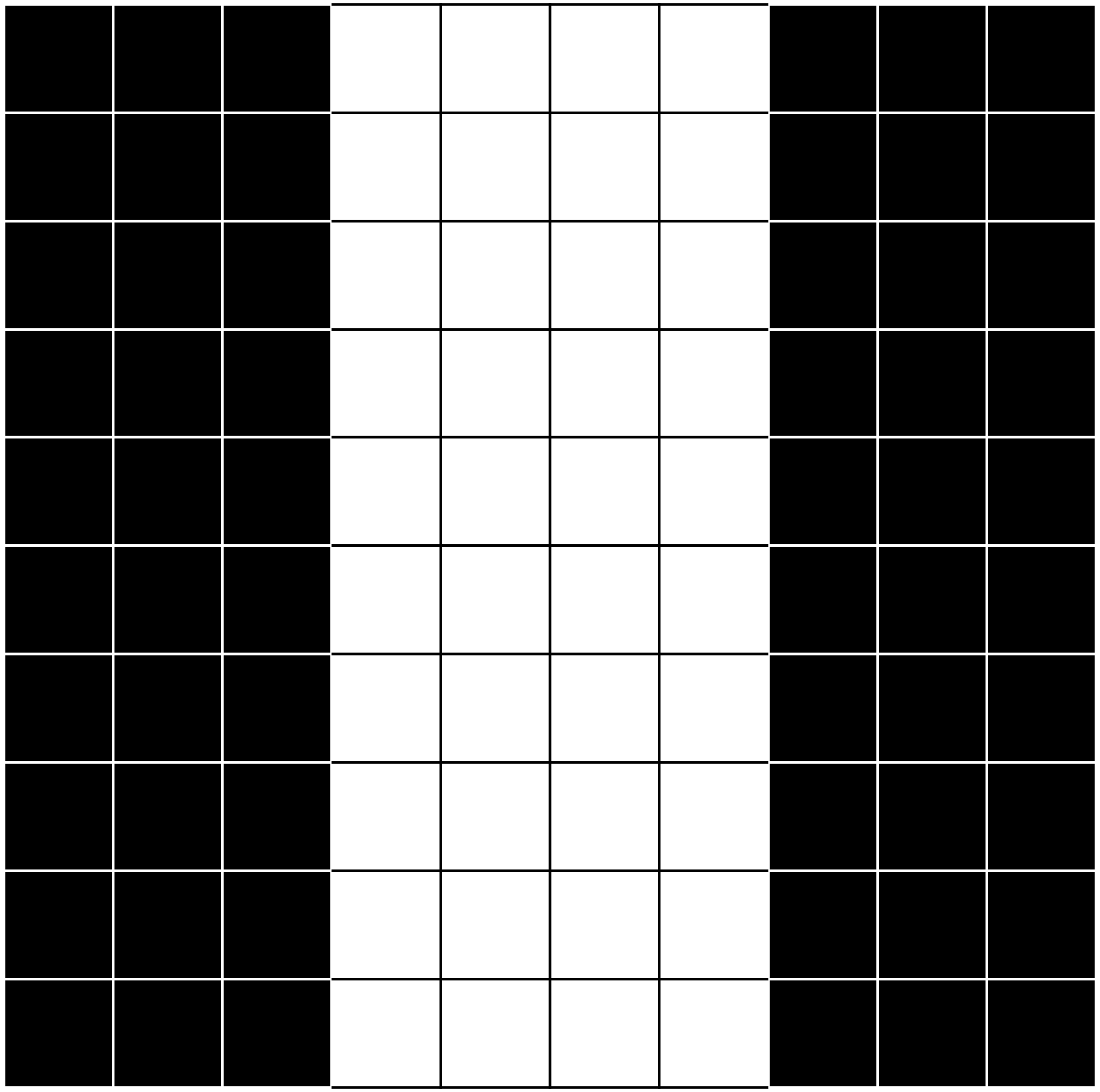


Convolutional Neural Networks



Convolution

X



W

-1	0	1
-1	0	1
-1	0	1

Convolution

$\sum w_{ij} x_{uv}$

X

0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0

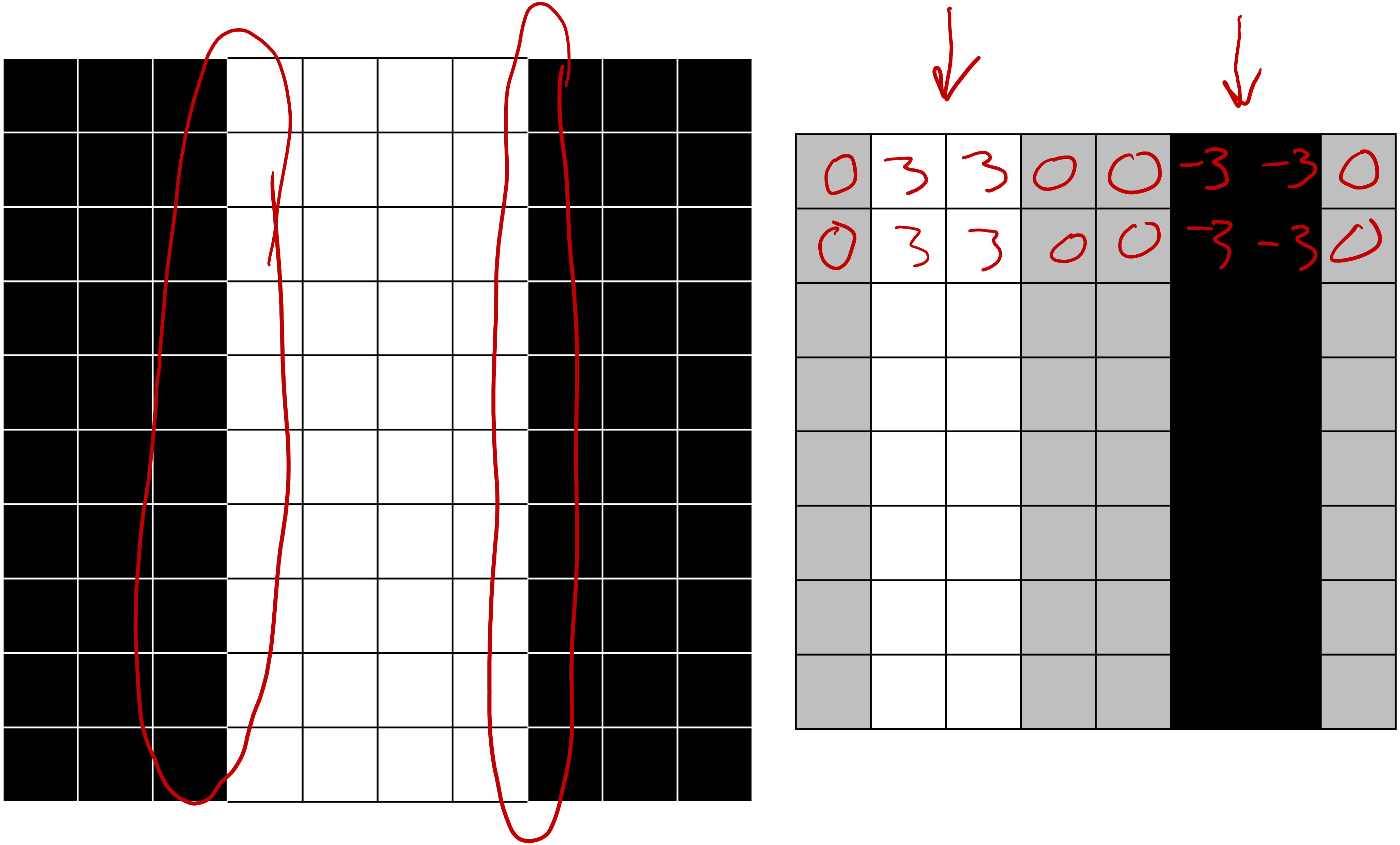
0 3 3 0

0

W

-1	0	1
-1	0	1
-1	0	1

Convolution



-1	0	1
-1	0	1
-1	0	1

Convolution

$\sum x_i w_i$

Signal processing definition

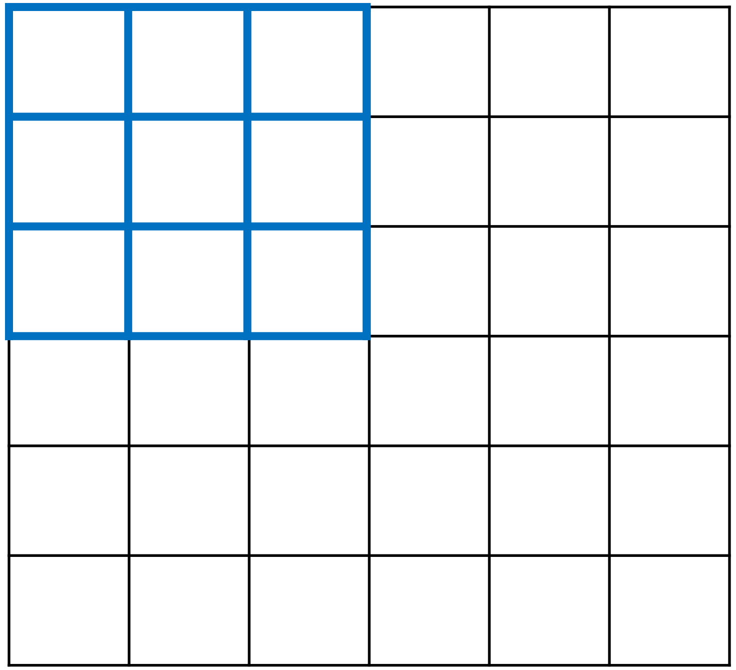
$$z[i,j] = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} x[i-u, j-v] \cdot w[u,v]$$

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

Relaxed definition

- Drop infinity; don't flip kernel

$$z[i,j] = \sum_{u=0}^{K-1} \sum_{v=0}^{K-1} x[i+u, j+v] \cdot w[u,v]$$



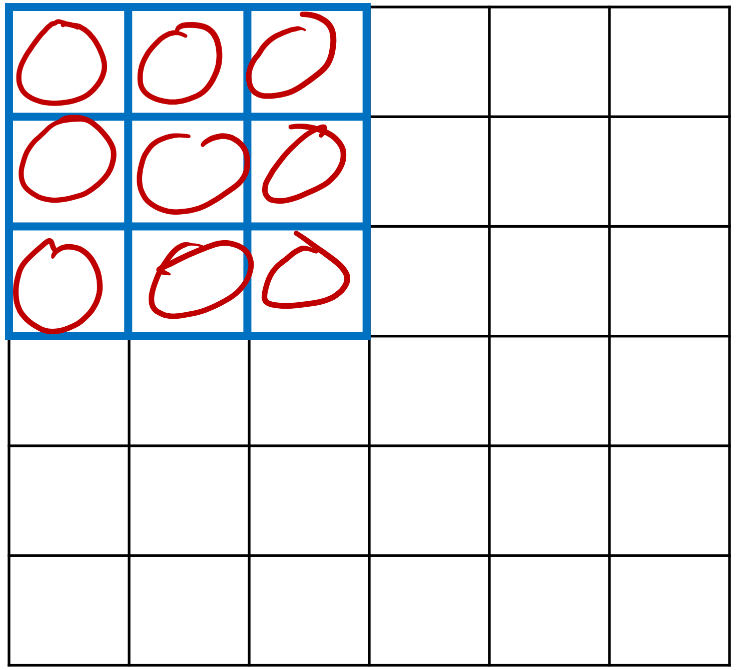
Convolution

Relaxed definition

$$z[i,j] = \sum_{u=0}^{K-1}\sum_{v=0}^{K-1} x[i+u, j+v] \cdot w[u,v]$$

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

```
for i in range(0, im_width - K + 1):
  for j in range(0, im_height - K + 1):
    im_out[i,j] = 0
    for u in range(0, K):
      for v in range(0, K):
        im_out[i,j] += im[i+u, j+v] * kernel[u,v]
```



GPU!!

Convolution: Padding

$$\boxed{\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}}$$

0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0

0	2	2	0	0	-2	-2	0
0	3	3	0	0	-3	-3	0
0	3	3	0	0	-3	-3	0
0	3	3	0	0	-3	-3	0
0	3	3	0	0	-3	-3	0
0	3	3	0	0	-3	-3	0
0	3	3	0	0	-3	-3	0
0	2	2	0	0	-2	-2	0

Piazza Poll 3 : Which kernel goes with which output image?

Input

X

K1

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

K2

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

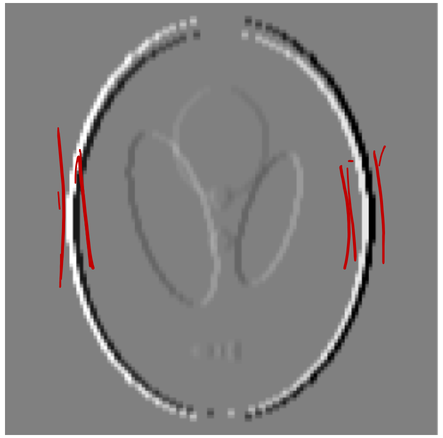
K3

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

Im1

Im2

Im3



Piazza Poll 3: Which kernel goes with which output image?

Input

K1

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

K2

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

K3

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

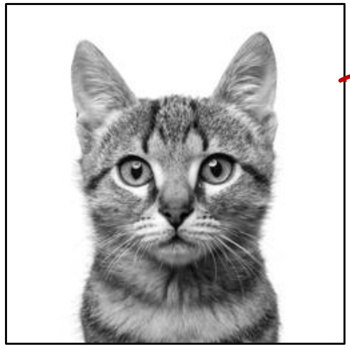
Im1

Im2

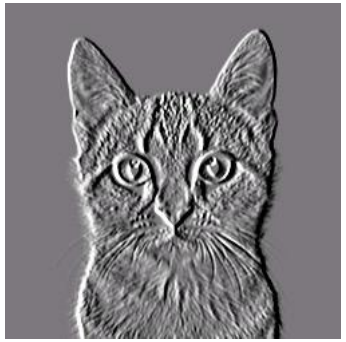
Im3

Convolutional Neural Networks

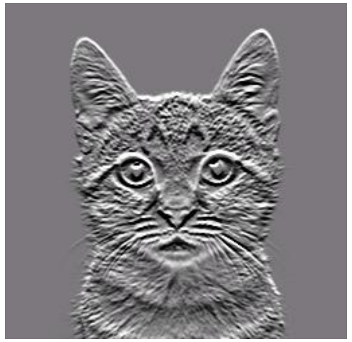
Convolution



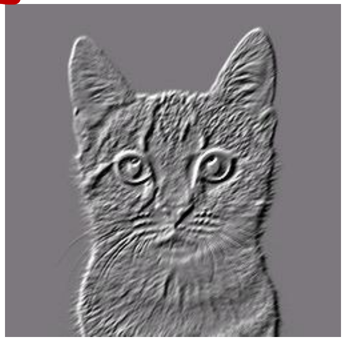
-1	0	1
-2	0	2
-1	0	1

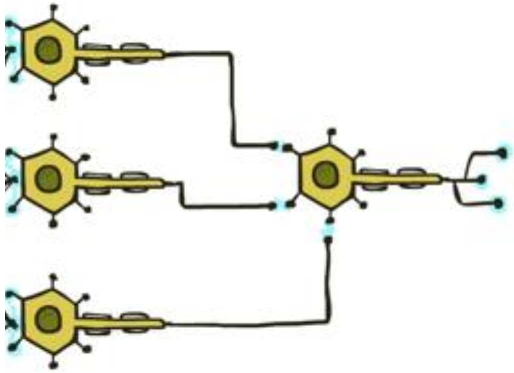


-1	-2	-1
0	0	0
1	2	1



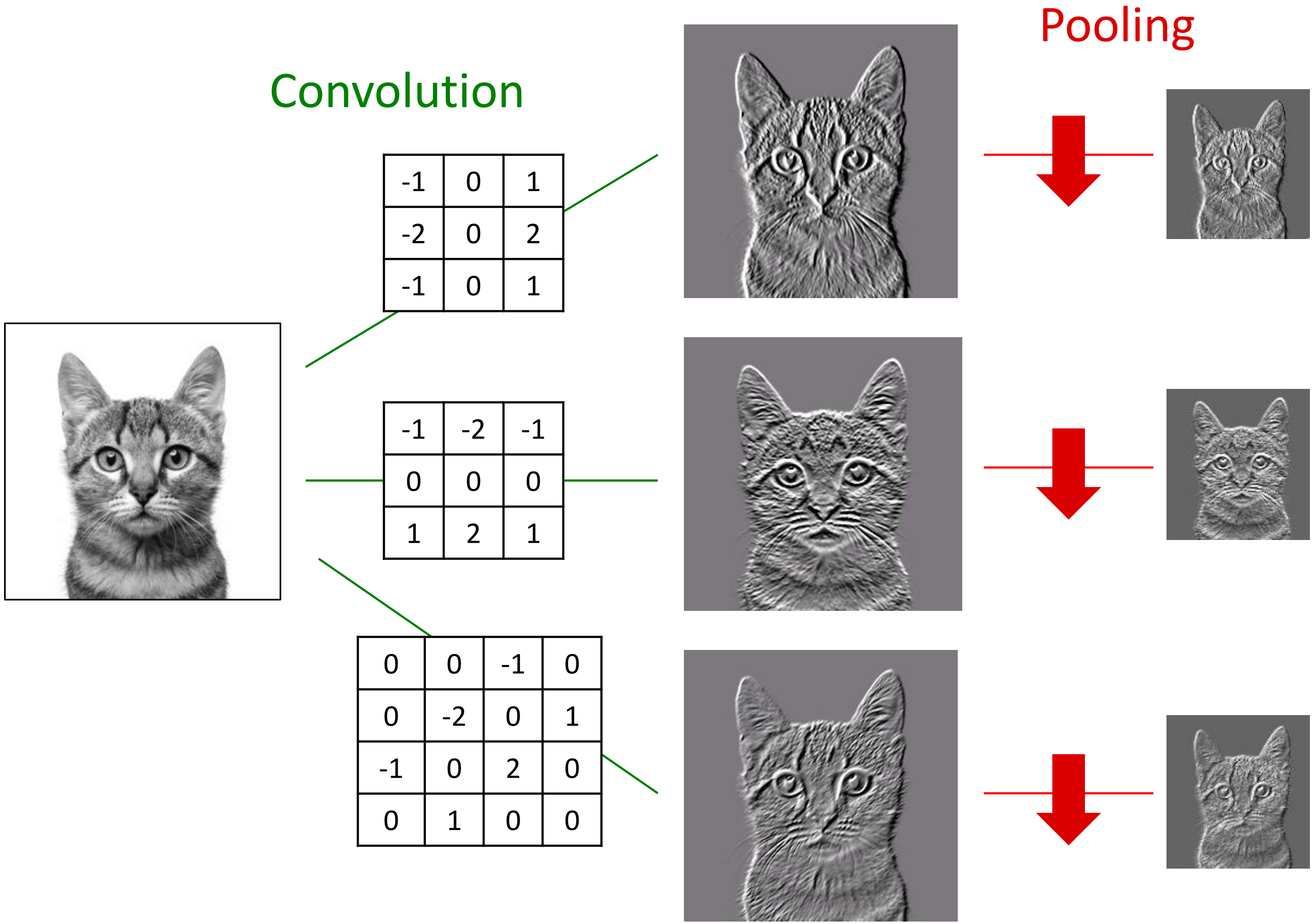
0	0	-1	0
0	-2	0	1
-1	0	2	0
0	1	0	0





CAT

Convolutional Neural Networks



Convolution: Stride=2

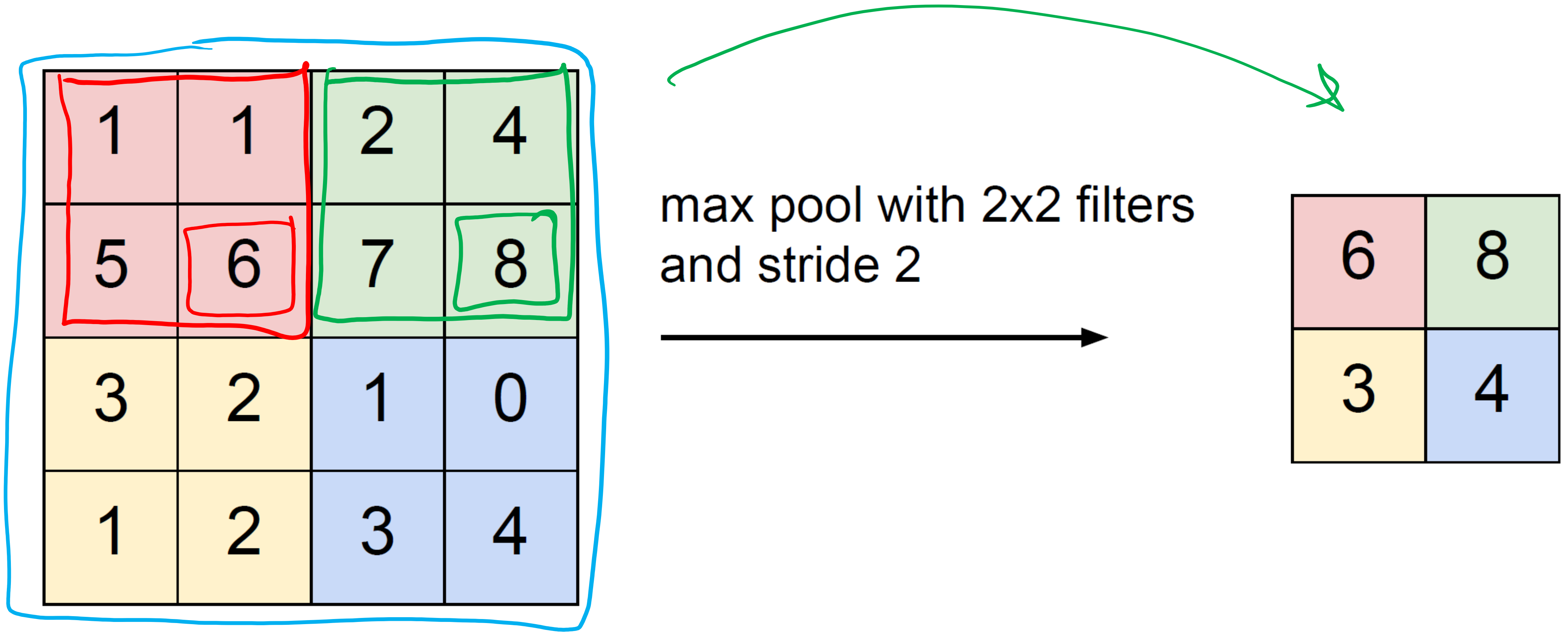
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0

average kernel

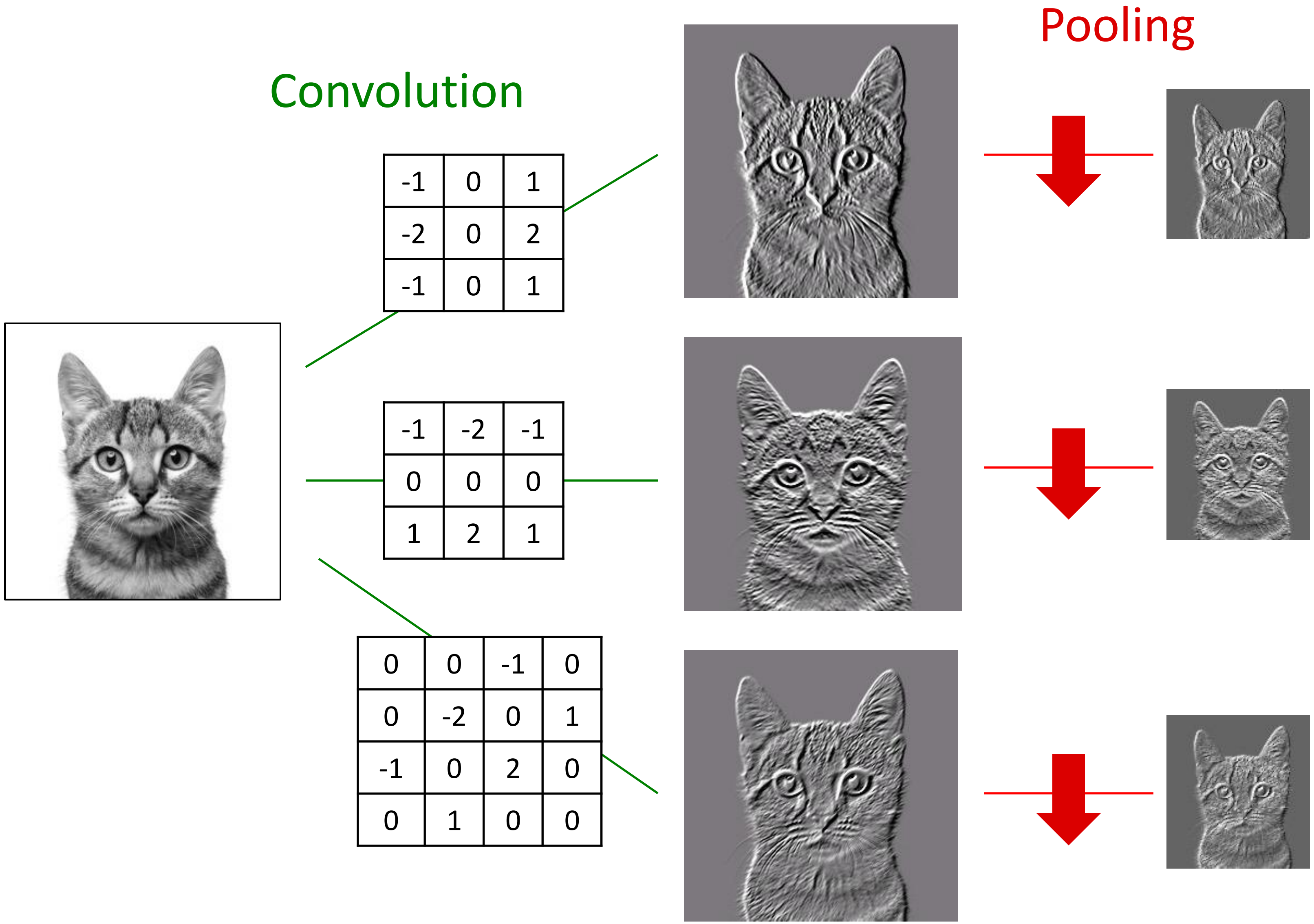
.25	.25
.25	.25

0 2 4 2 0
0 2 4

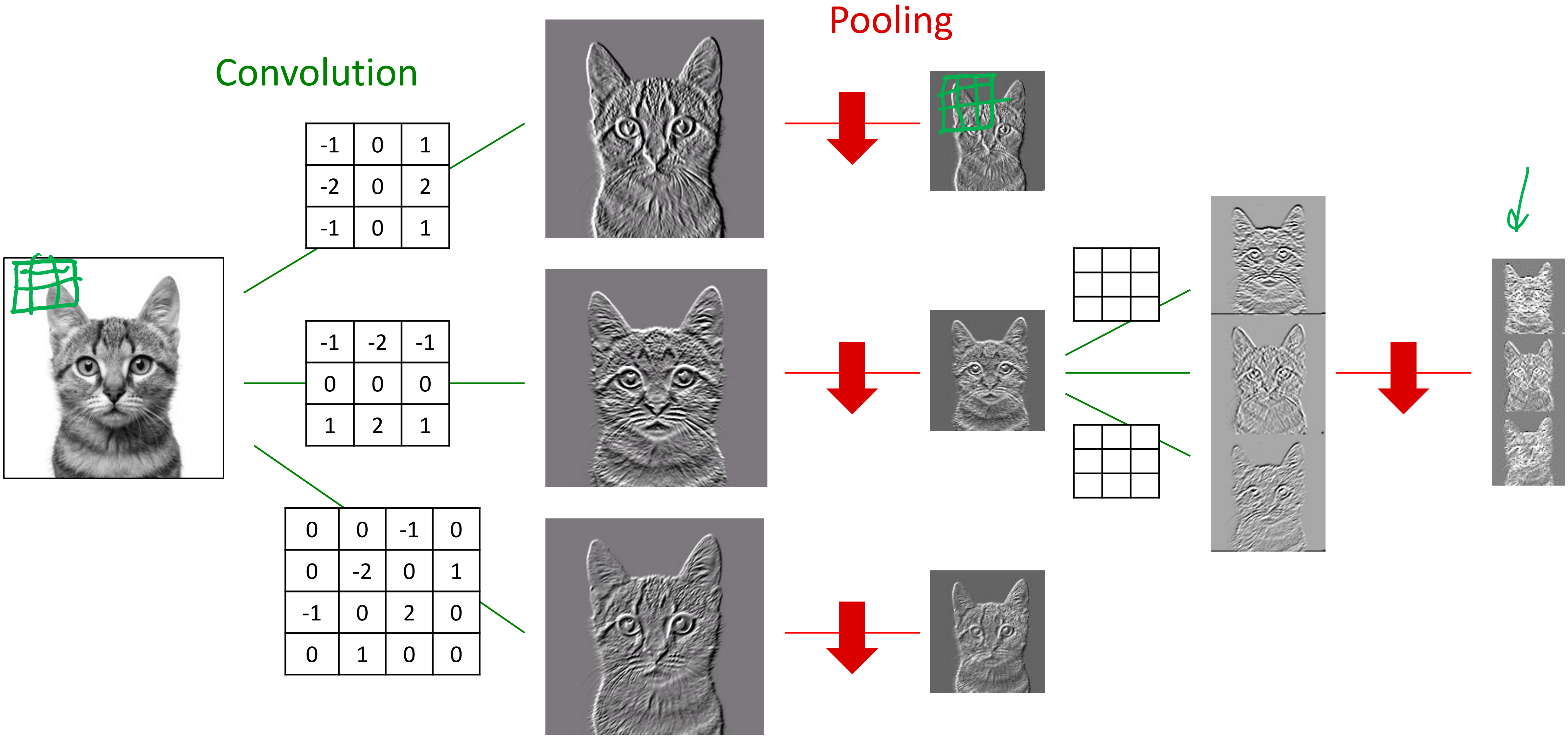
Stride: Max Pooling



Convolutional Neural Networks



Convolutional Neural Networks



Convolutional Neural Networks

2 @ 3x3
1 @ 4x4

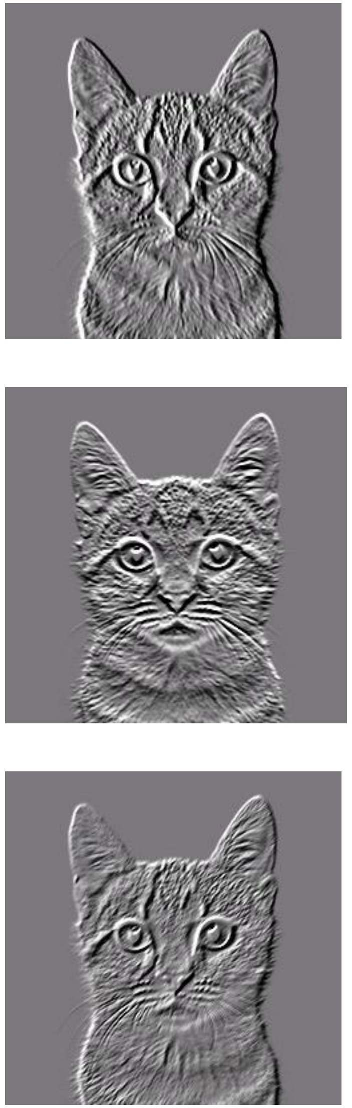
Convolution

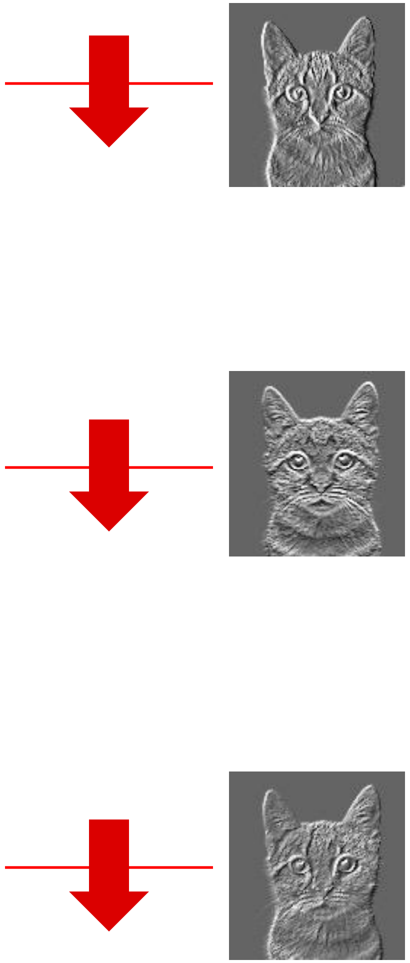
Pooling

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

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

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





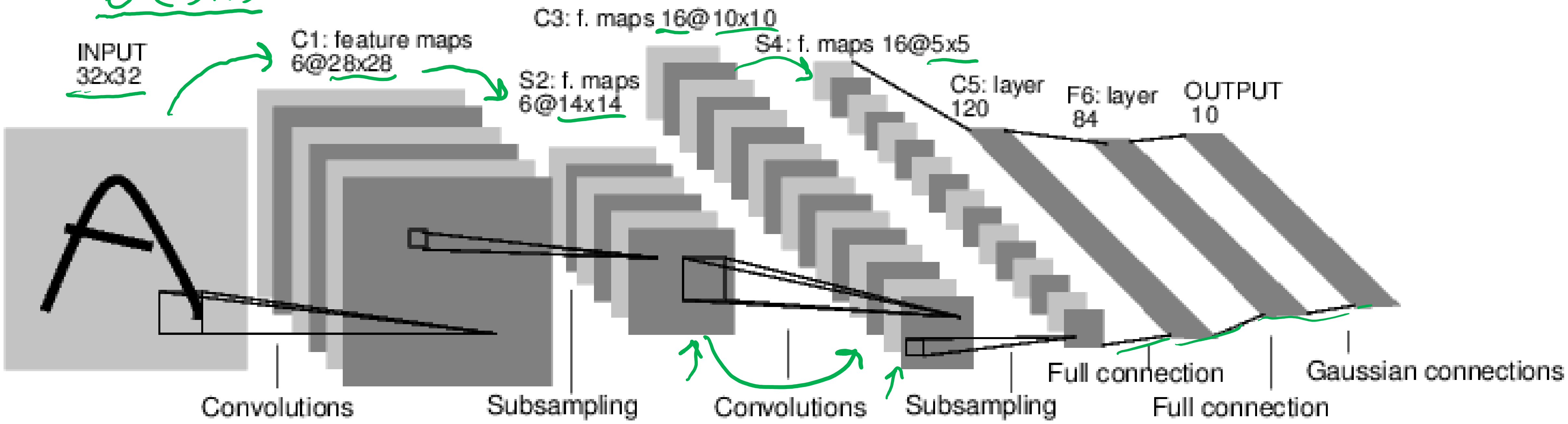
Convolutional Neural Networks

Lenet5 – Lecun, et al, 1998

- Convnets for digit recognition

150

6 @5x5

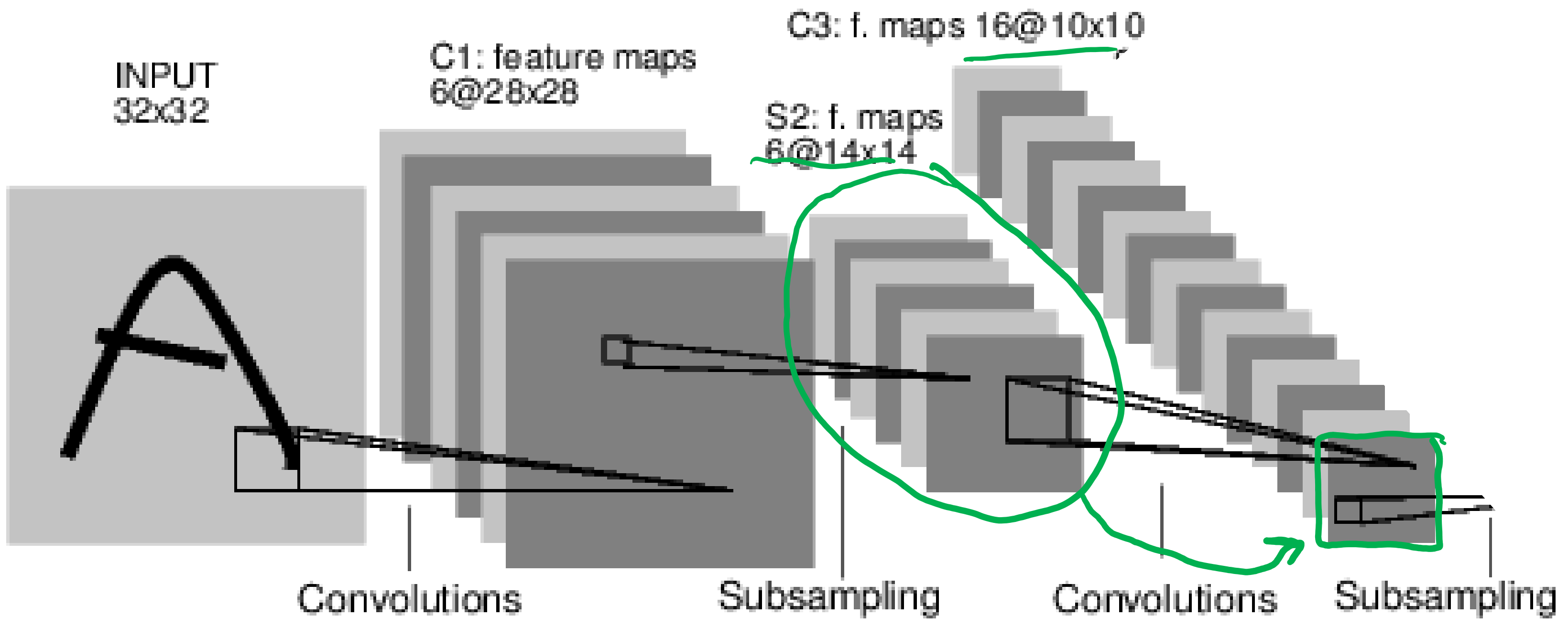


LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

Question:

How big many convolutional weights between S2 and C3?

- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10

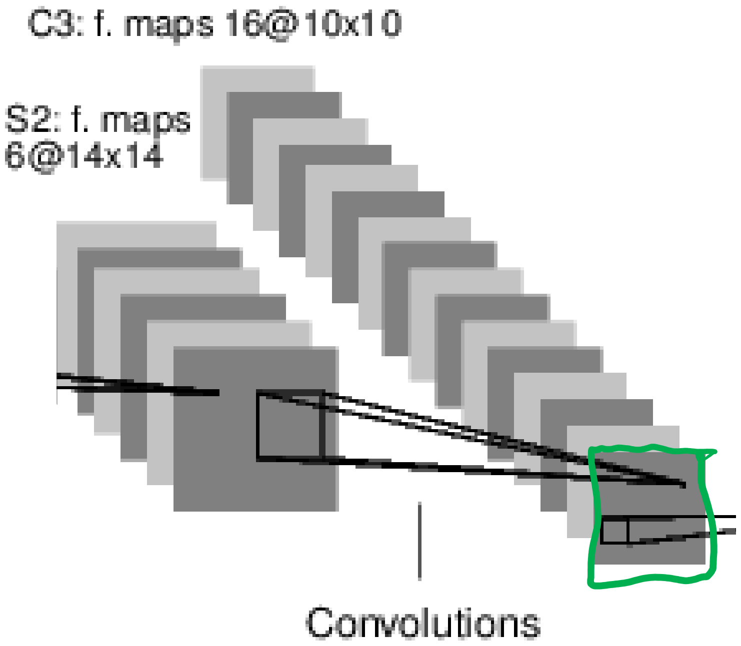


Question:

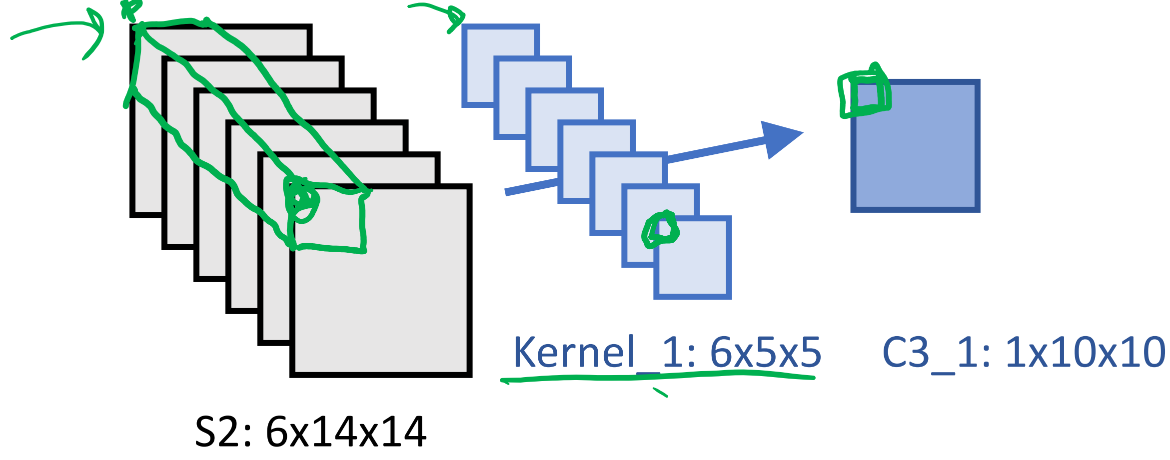
How big many convolutional weights between S2 and C3?

- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10

$$\sum w_{ijk} x_{uvz}$$



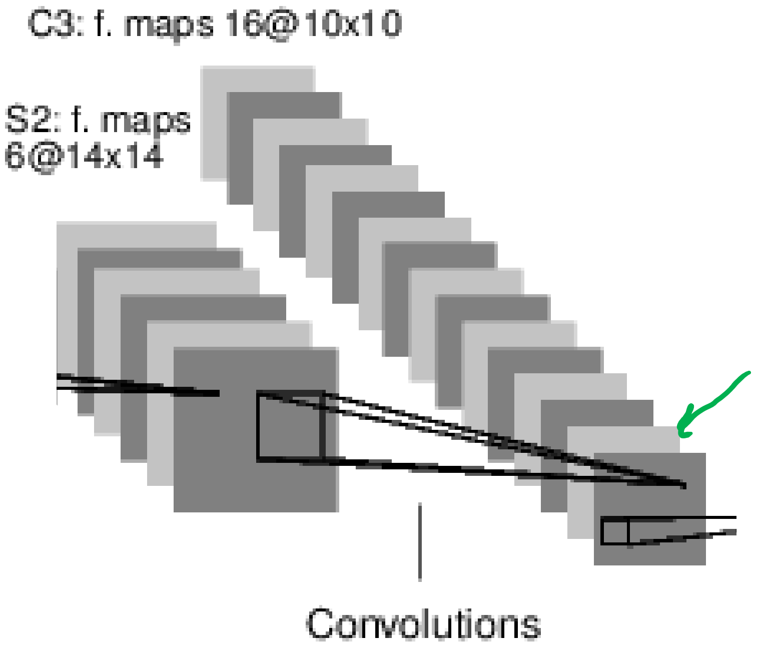
One image in C3 is actually the result of a 3D convolution



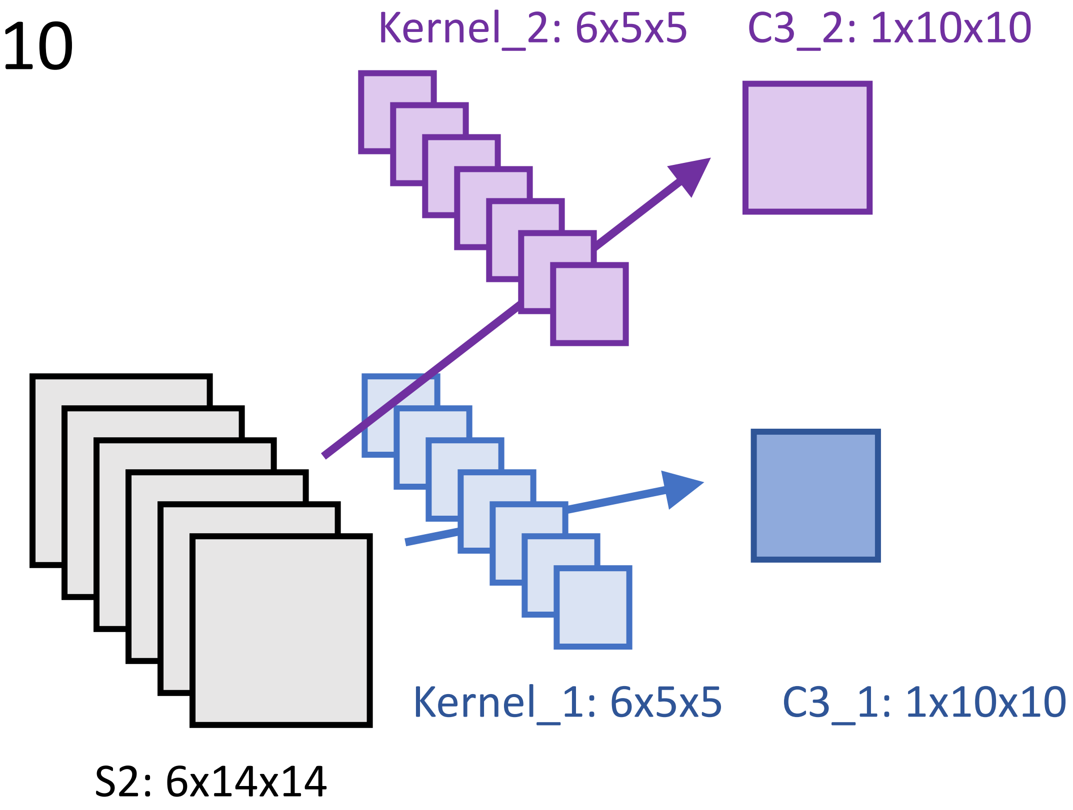
Question:

How big many convolutional weights between S2 and C3?

- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10



Each image in C3 convolved S2
convolved with a different 3D kernel

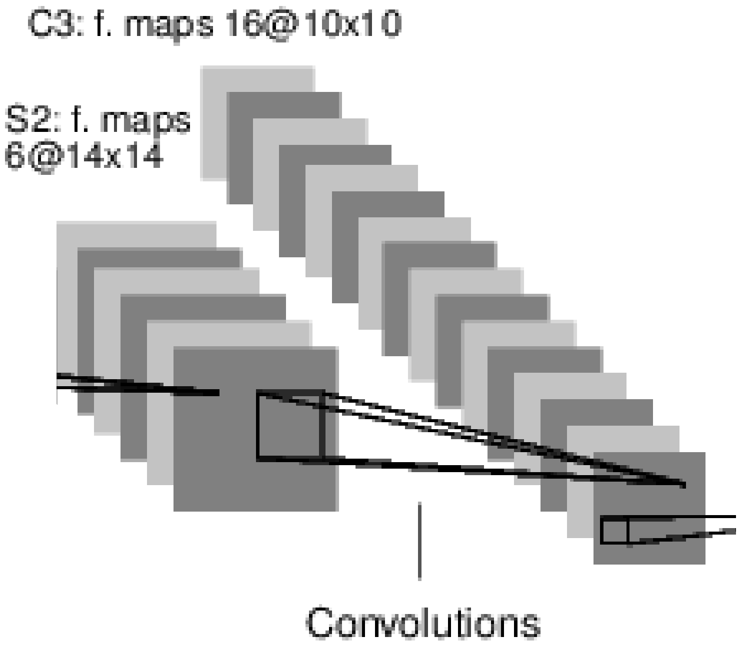


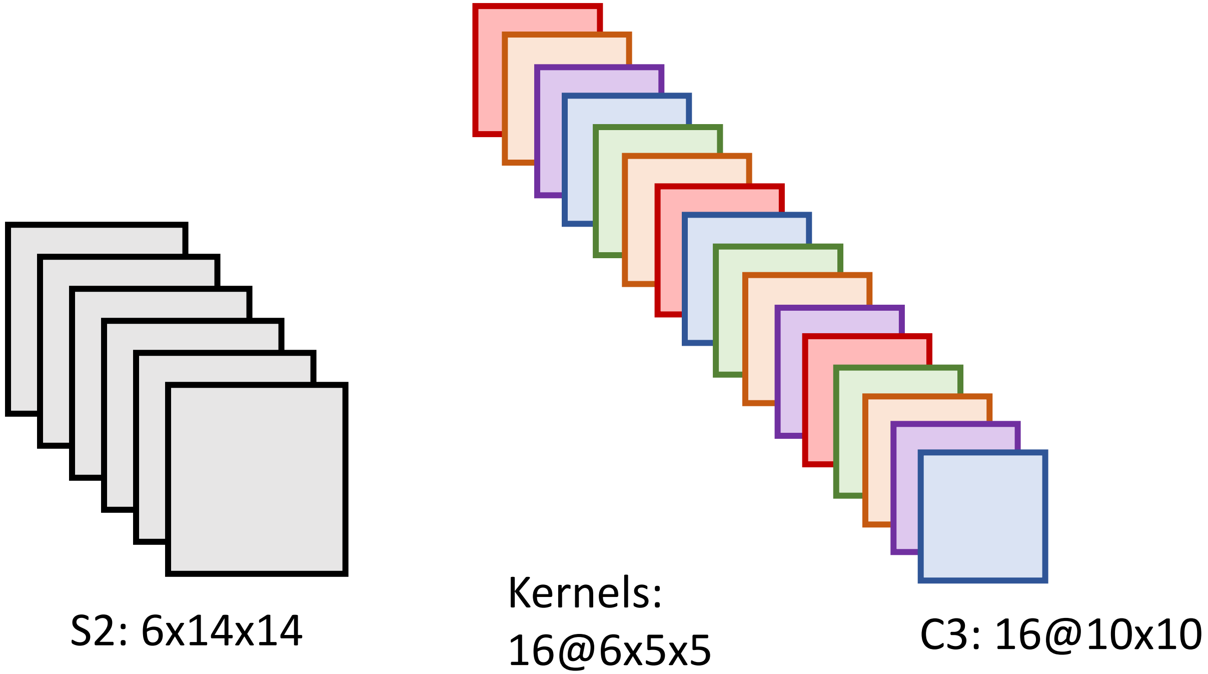
Question:

How big many convolutional weights between S2 and C3?

- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10

The 16 images in C3 are the result of doing 16 3D convolutions of S2 with 16 different 6x5x5 kernels. Assuming no bias term, this is 16x6x5x5 weights!

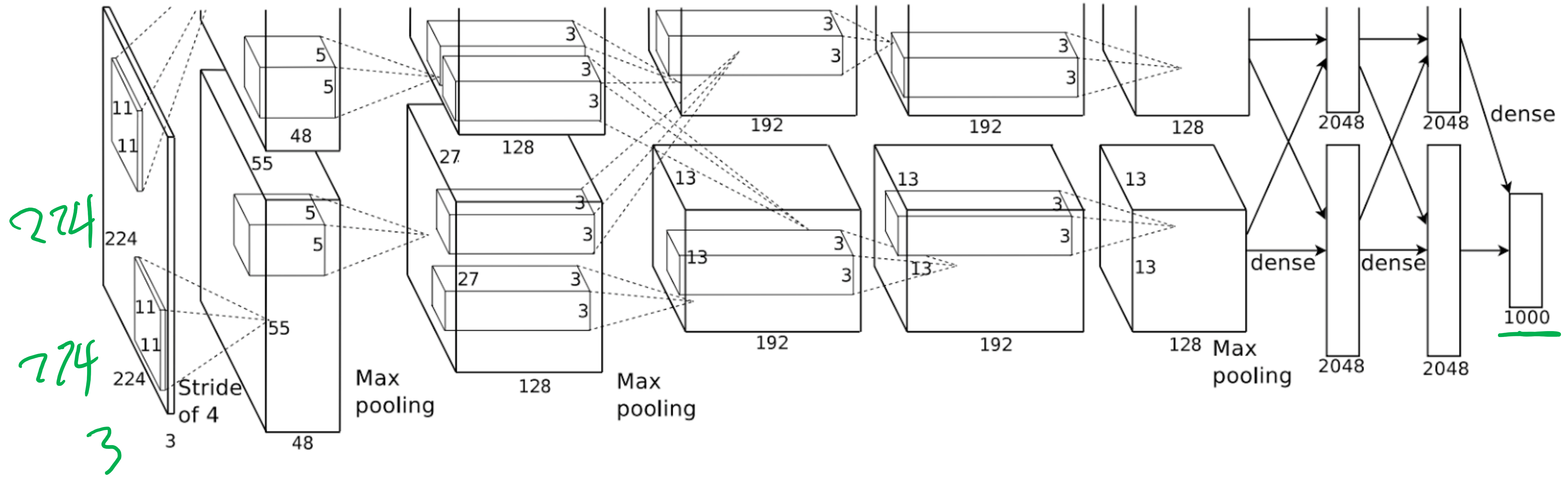




Convolutional Neural Networks

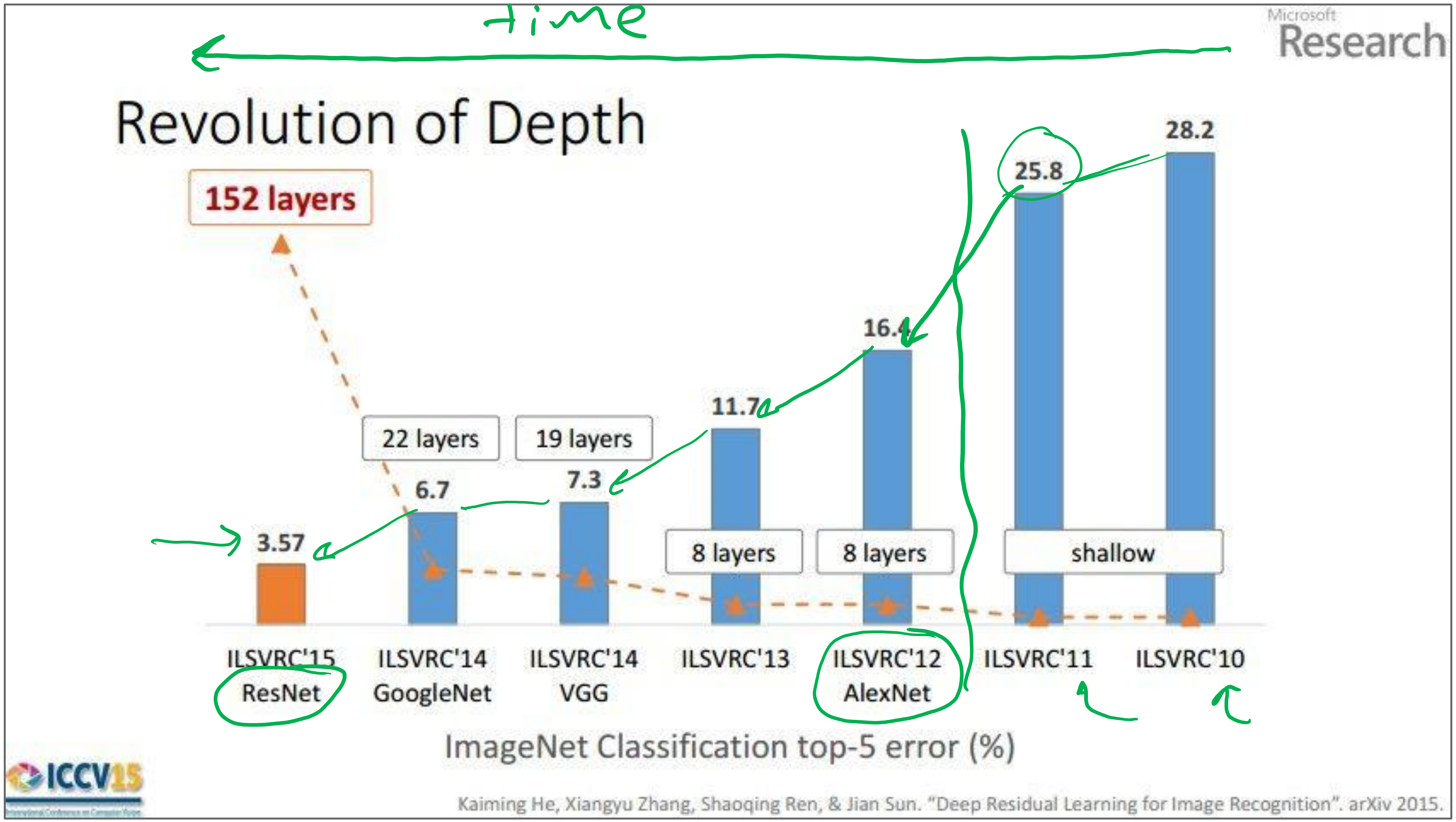
Alexnet – Lecun, et al, 2012

- Convnets for image classification
- More data & more compute power



Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." NIPS, 2012.

CNNs for Image Recognition



Slide from Kaiming He