Announcements

Assignments:

- HW4
  - Due tonight!
- HW5
  - Due date Thu, 2/27, 11:59 pm

Midterm

- See Piazza post for details

Canvas updated

- Participation points

Plan

Last time

- Neural Networks
  - Universal Approximation
  - Optimization / Backpropagation

Today

- Wrap-up Optimization / Backpropagation
- Course Survey
- Convolutional Neural Networks

Introduction to Machine Learning

Neural Networks

Instructor: Pat Virtue

Reminder: Calculus Chain Rule (scalar version)

$$y = f(z)$$
$$z = g(x)$$

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

Network Optimization

$$J(\mathbf{w}) = z_3$$

$$z_3 = f_3(w_3, z_2)$$

$$z_2 = f_2(w_2, z_1)$$

$$z_1 = f_1(w_1, x)$$

$$\frac{\partial J}{\partial w_3} = \frac{\partial J}{\partial z_3}\frac{\partial z_3}{\partial w_3}$$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial z_3}\frac{\partial z_3}{\partial z_2}\frac{\partial z_2}{\partial w_2}$$

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial z_3}\frac{\partial z_3}{\partial z_2}\frac{\partial z_2}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

Lots of repeated calculations

Backpropagation (so-far)

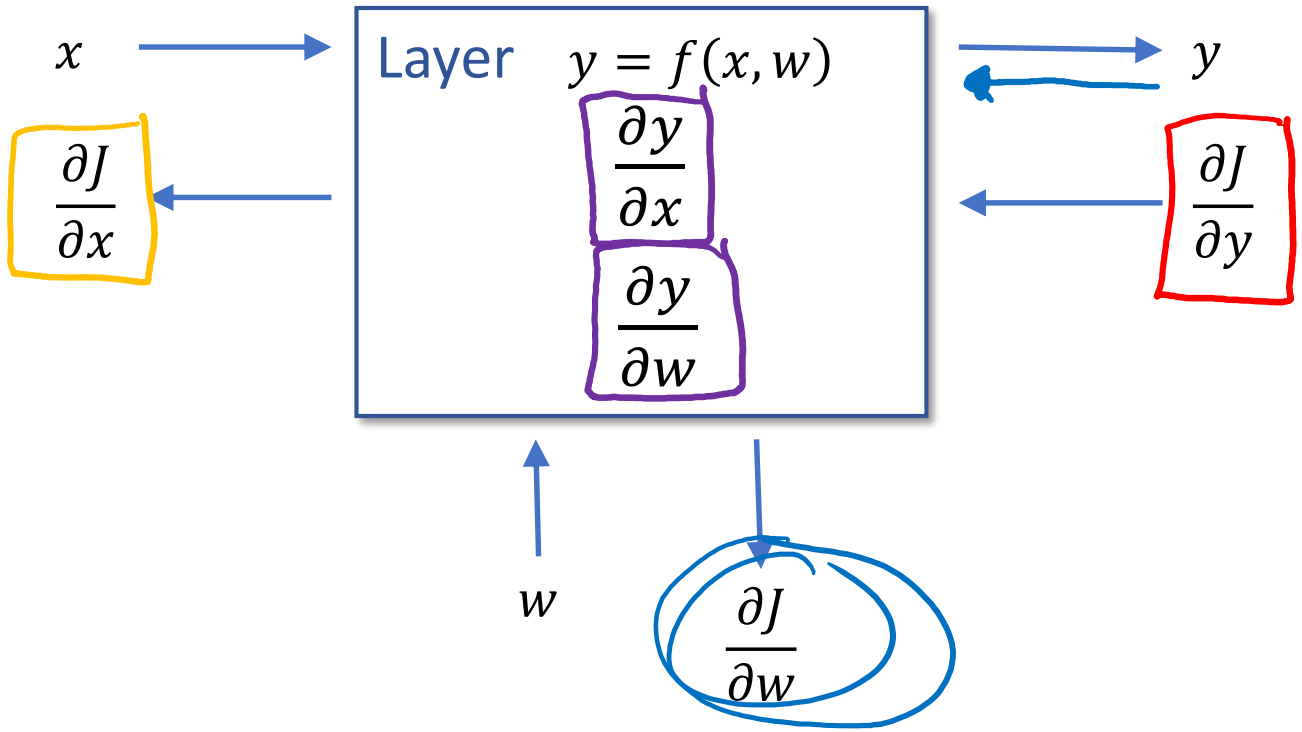
Compute derivatives per layer, utilizing previous derivatives

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

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

Need:

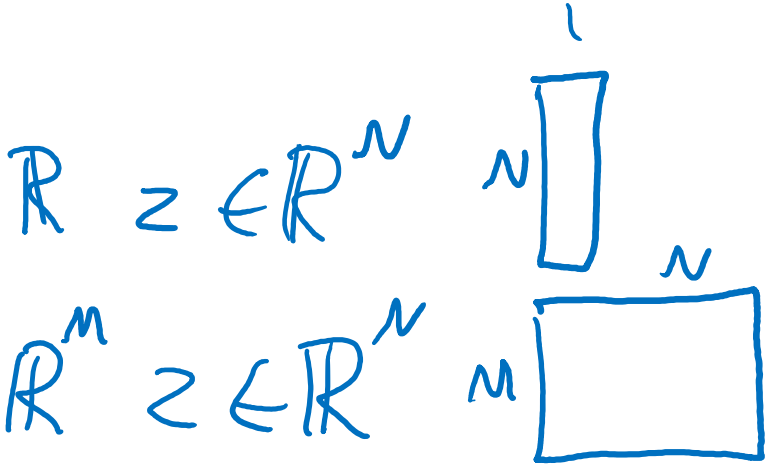
- $\frac{\partial J}{\partial x} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial x}$
- $\frac{\partial J}{\partial w} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial w}$



Reminder: Matrix Calculus

Gradient, Jacobian, etc

$\nabla_z f_1$ Gradient $f_1 \in \mathbb{R}$ $z \in \mathbb{R}^N$



$\frac{\partial f_2}{\partial \vec{z}}$ Jacobian $f_2 \in \mathbb{R}^M$ $z \in \mathbb{R}^N$

$\frac{\partial f_1}{\partial z}$

$\nabla_{\vec{z}} f_1^T = \frac{\partial f_1}{\partial \vec{z}}$

Calculus Chain Rule

Scalar:

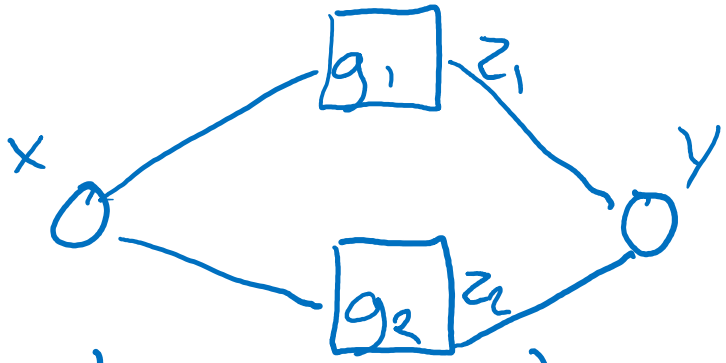
$$y = f(z)$$
$$z = g(x)$$
$$\frac{dy}{dx} = \frac{dy}{dz}\frac{dz}{dx}$$

Multivariate:

$$\mathbf{y} = f(\mathbf{z})$$
$$\mathbf{z} = g(x)$$
$$\frac{dy}{dx} = \sum_j \frac{\partial y}{\partial z_j}\frac{\partial z_j}{\partial x}$$

Multivariate:

$$\mathbf{y} = f(\mathbf{z})$$
$$\mathbf{z} = g(\mathbf{x})$$
$$\frac{dy_i}{dx_k} = \sum_j \frac{\partial y_i}{\partial z_j}\frac{\partial z_j}{\partial x_k}$$



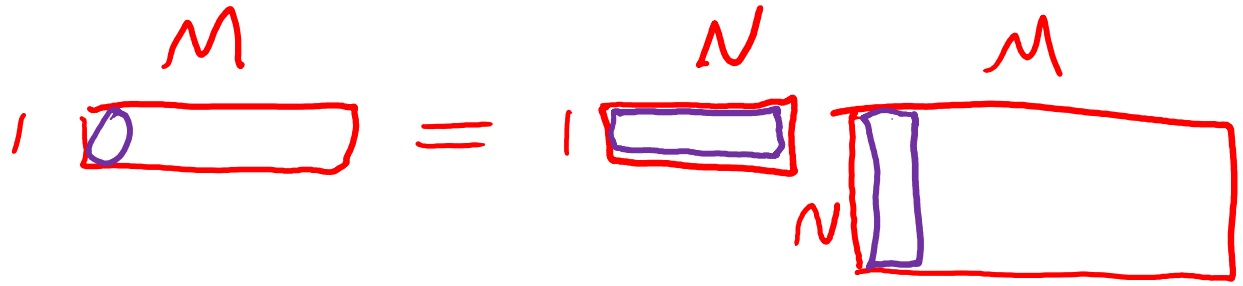
$$\frac{dy}{dx} = \frac{dy}{dz_1}\frac{dz_1}{dx} + \frac{dy}{dz_2}\frac{dz_2}{dx}$$

Piazza Poll 1:

$y = f(\boldsymbol{z})$ $\quad y \in \mathbb{R}$

$\boldsymbol{z} = g(\boldsymbol{x})$ $\quad \boldsymbol{x} \in \mathbb{R}^M \quad \boldsymbol{z} \in \mathbb{R}^N$

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



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

B. $\frac{\partial y}{\partial \boldsymbol{z}}^T \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$

C. $\frac{\partial y}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}^T$

D. $\frac{\partial y}{\partial \boldsymbol{z}}^T \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}^T$

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

F. None of the above

Network Optimization

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

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

$z_3 = f_3(w_C, z_1)$

$z_2 = f_2(w_B, z_1)$

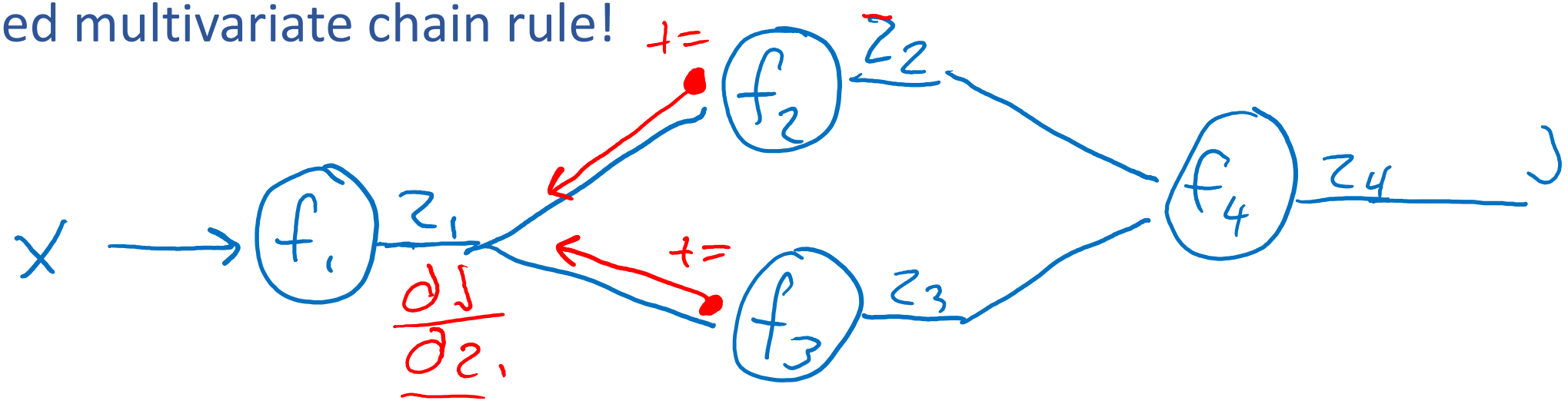
$z_1 = f_1(w_A, x)$

$\rightarrow \frac{\partial J}{\partial w_E}$

$\rightarrow \frac{\partial J}{\partial w_D}$ . . . . . $\rightarrow \frac{\partial J}{\partial w_A}$

0

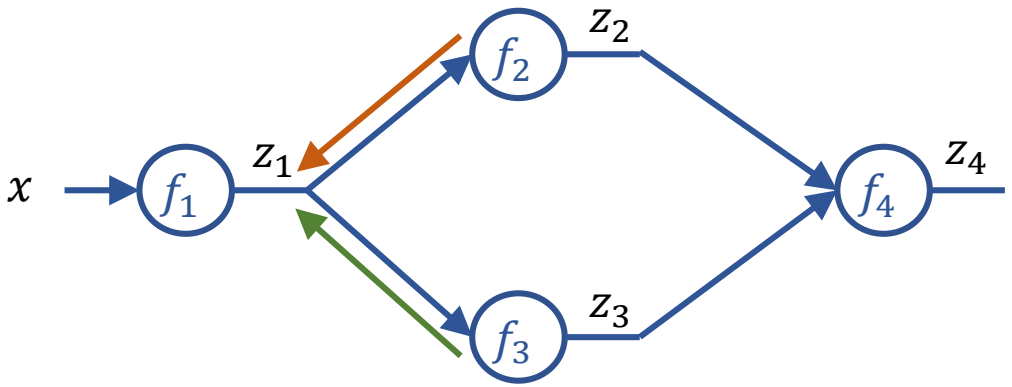
Need multivariate chain rule!



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: $J(\boldsymbol{w})$

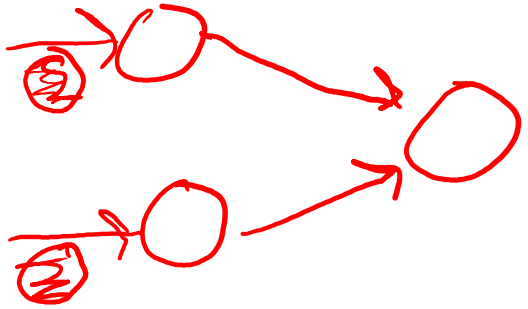
Arbitrary layer: $y = f(x, w)$

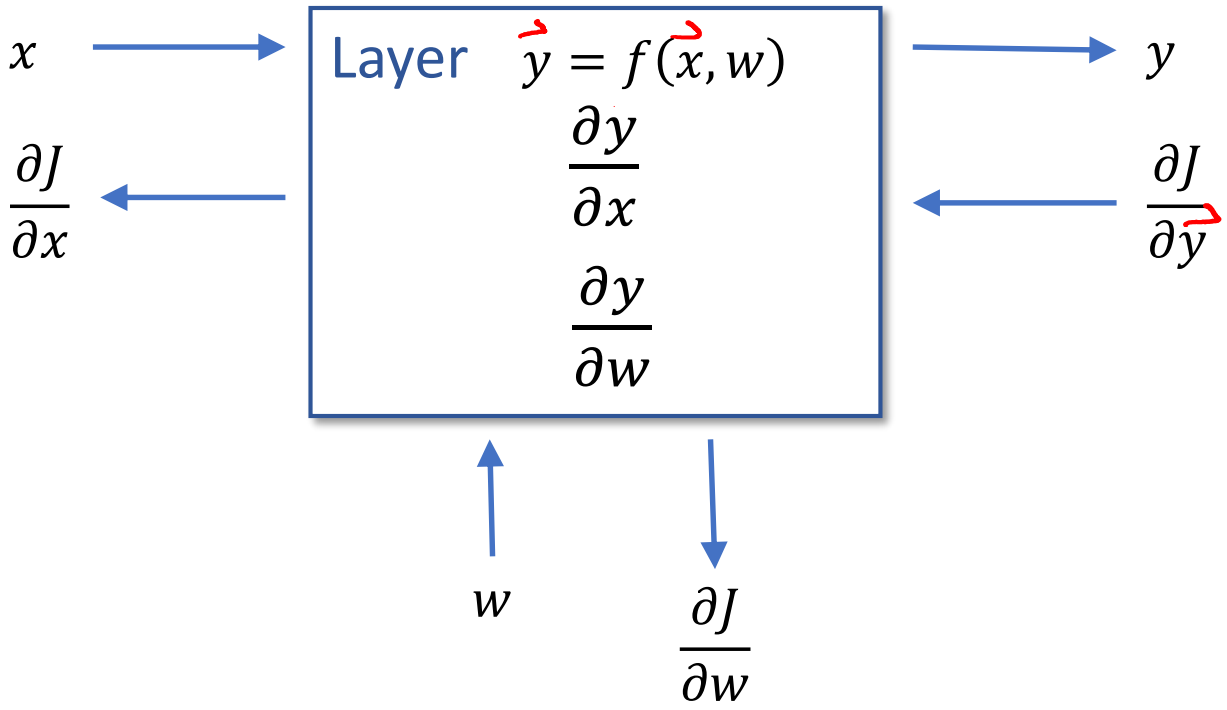
Init:

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

Compute:

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





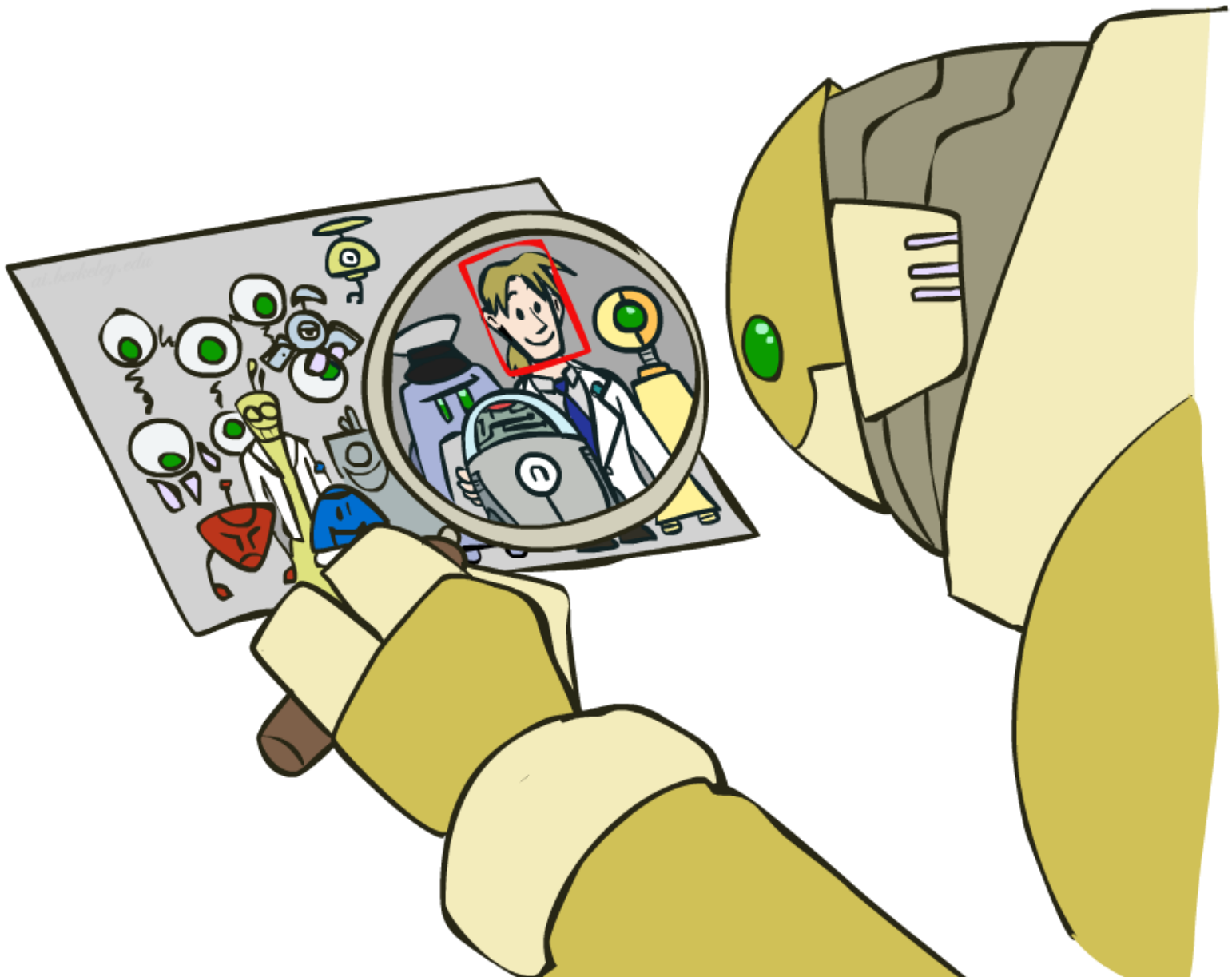
Course Survey

Introduction to Machine Learning

Convolutional Neural Networks

Instructor: Pat Virtue

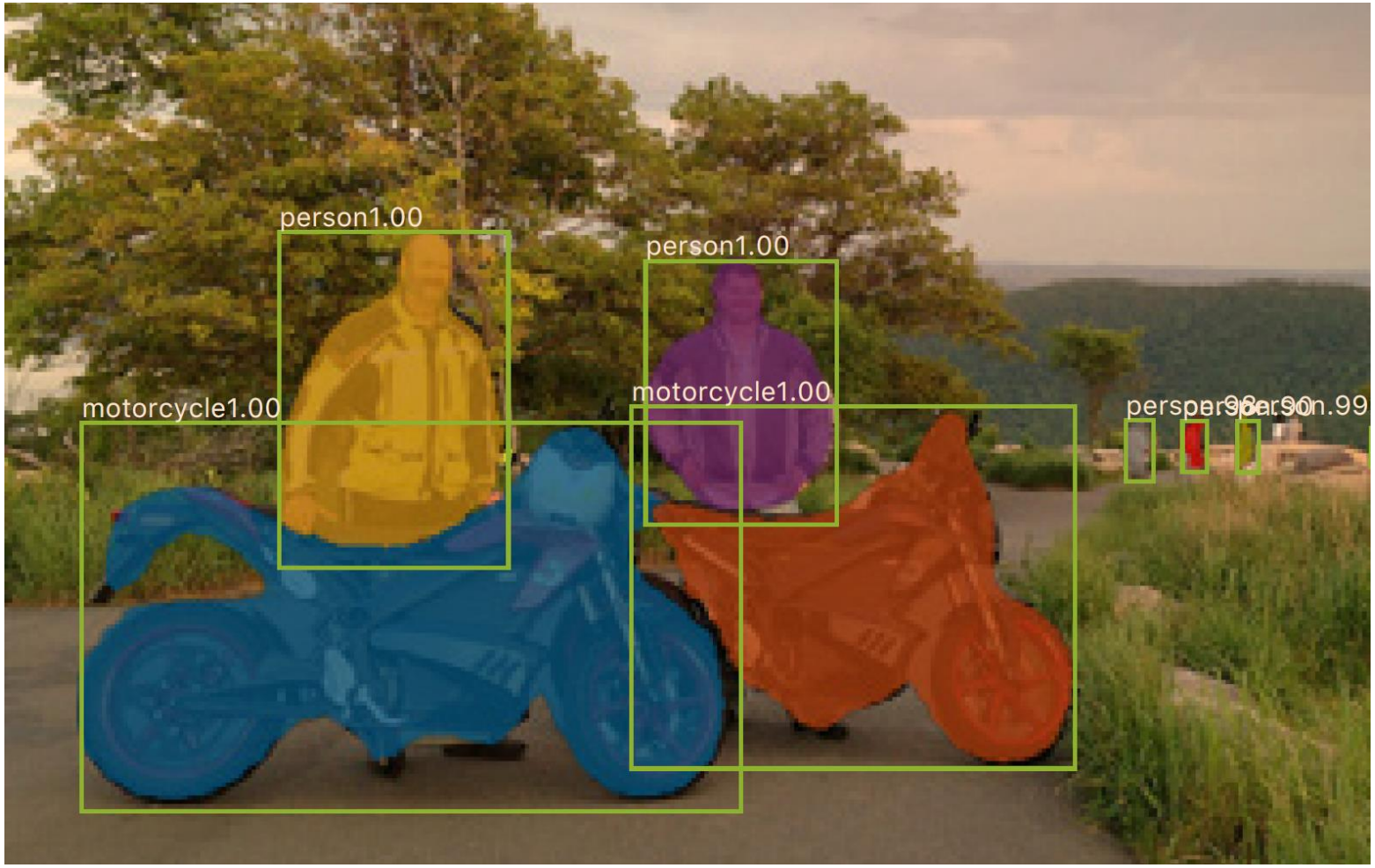
Computer Vision: How far along are we?



Computer Vision: How far along are we?

Terminator 2, 1991

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

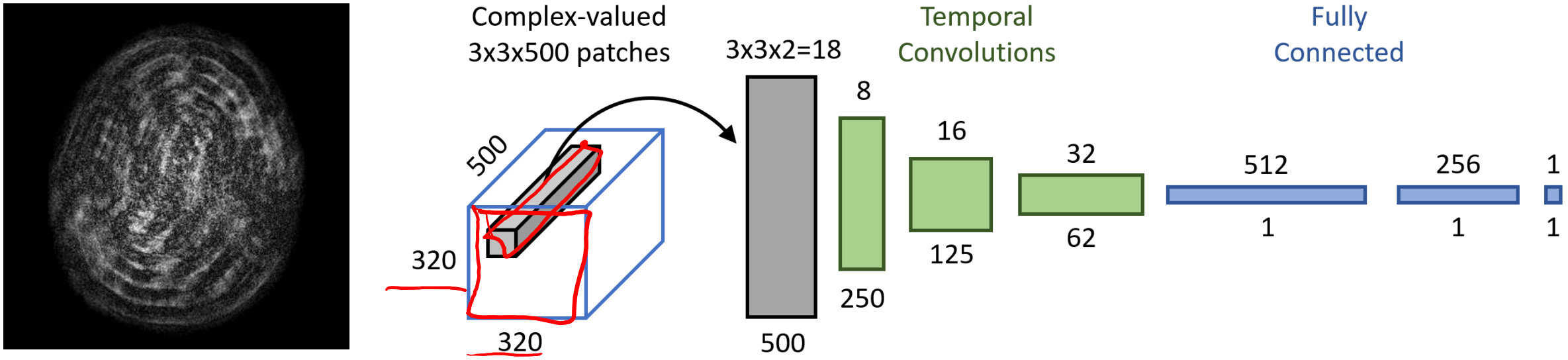
Computer Vision: Domain Transfer

CycleGAN

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

Temporal Convolution

MR Fingerprinting



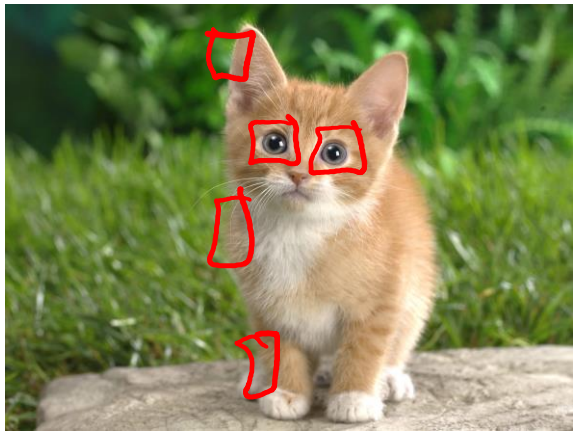
Patrick Virtue , Jonathan I Tamir , Mariya Doneva , Stella X Yu , and Michael Lustig. "Learning Contrast Synthesis from MR Fingerprinting", ISMRM 2018.

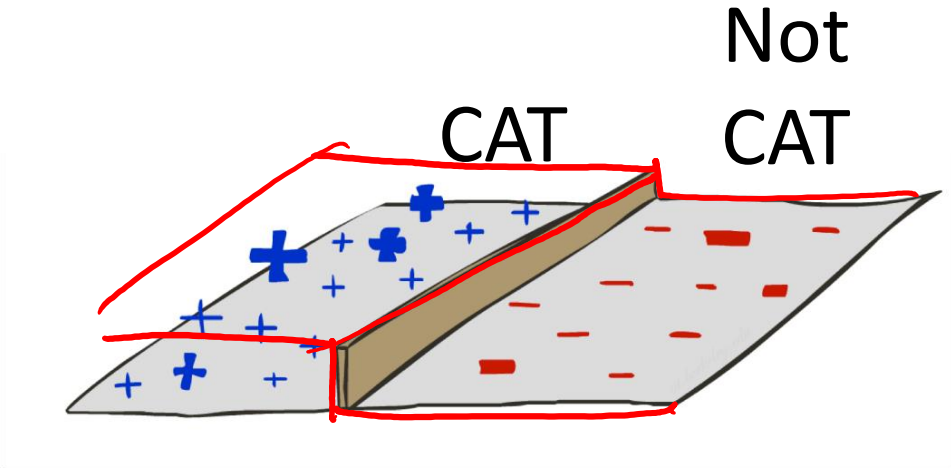
Outline

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

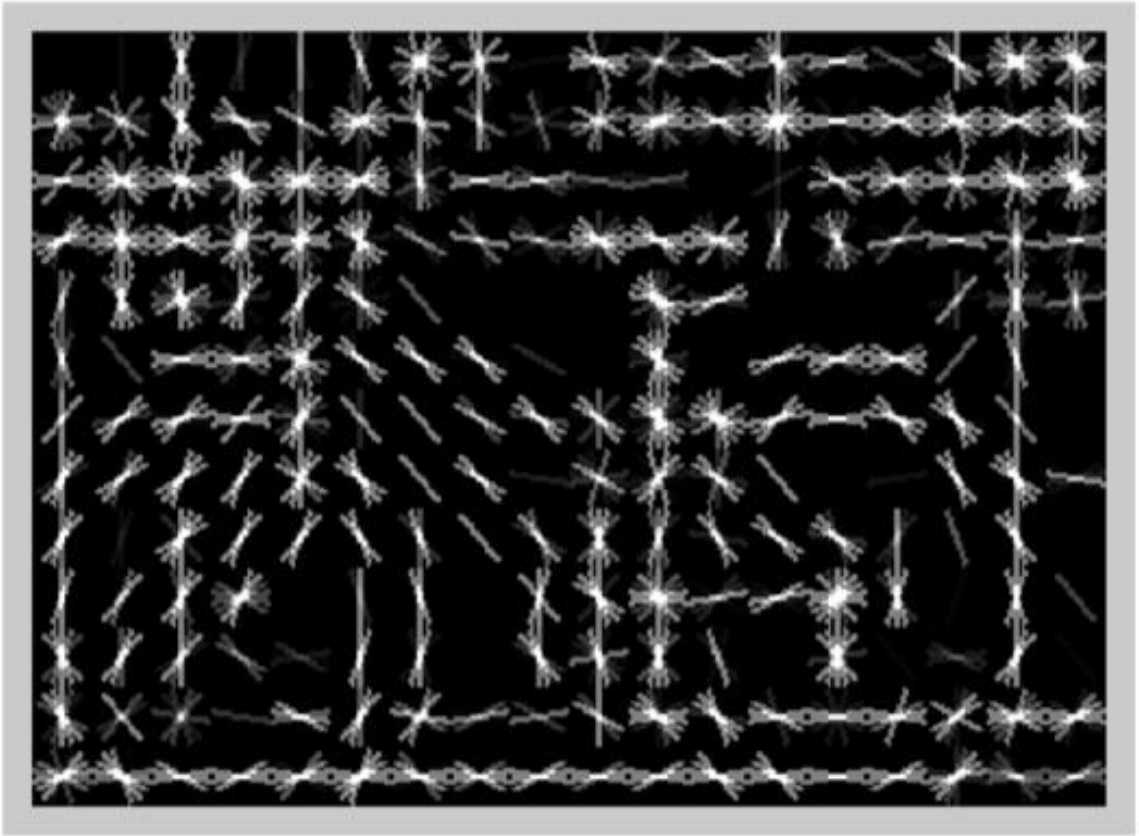
What's the problem with just directly classifying raw pixels in high dimensional space?





CAT

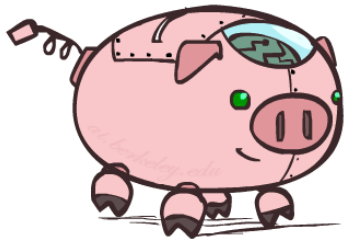
Image Classification

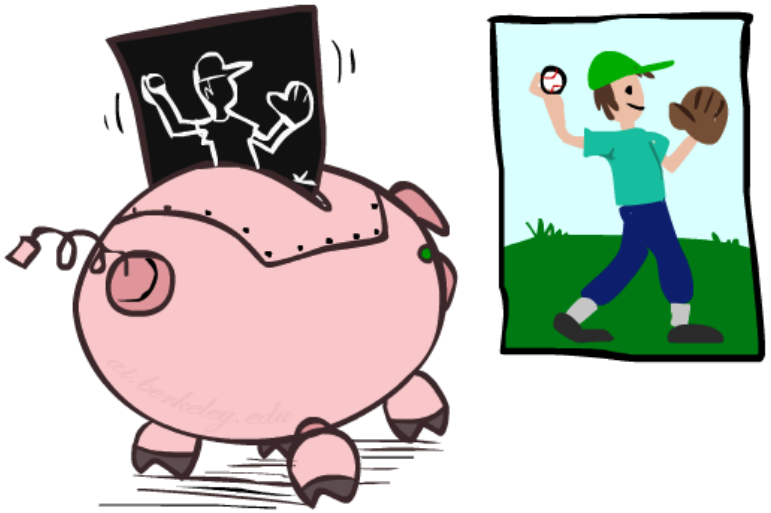


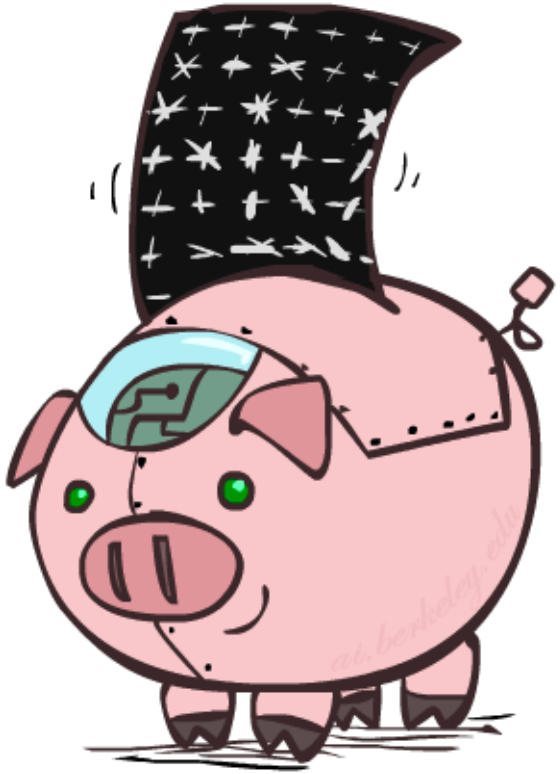
[Dalal and Triggs, 2005]

HoG Filter

HoG: Histogram of oriented gradients



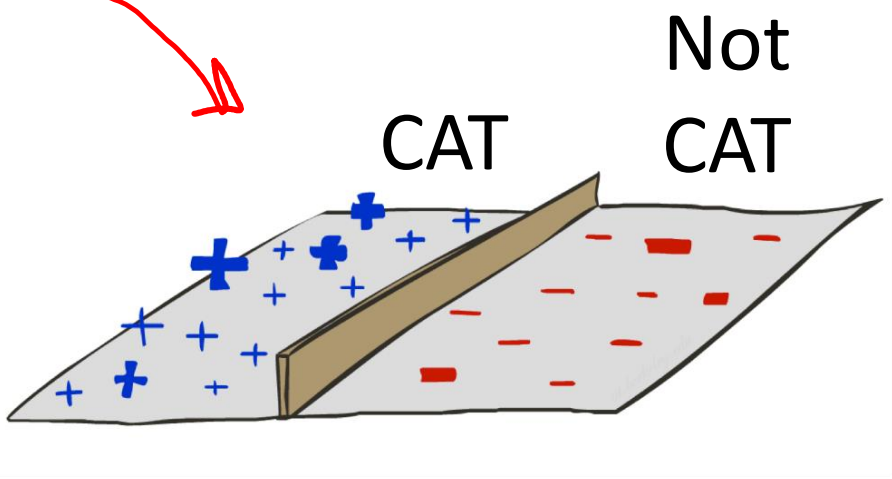




[Dalal and Triggs, 2005]

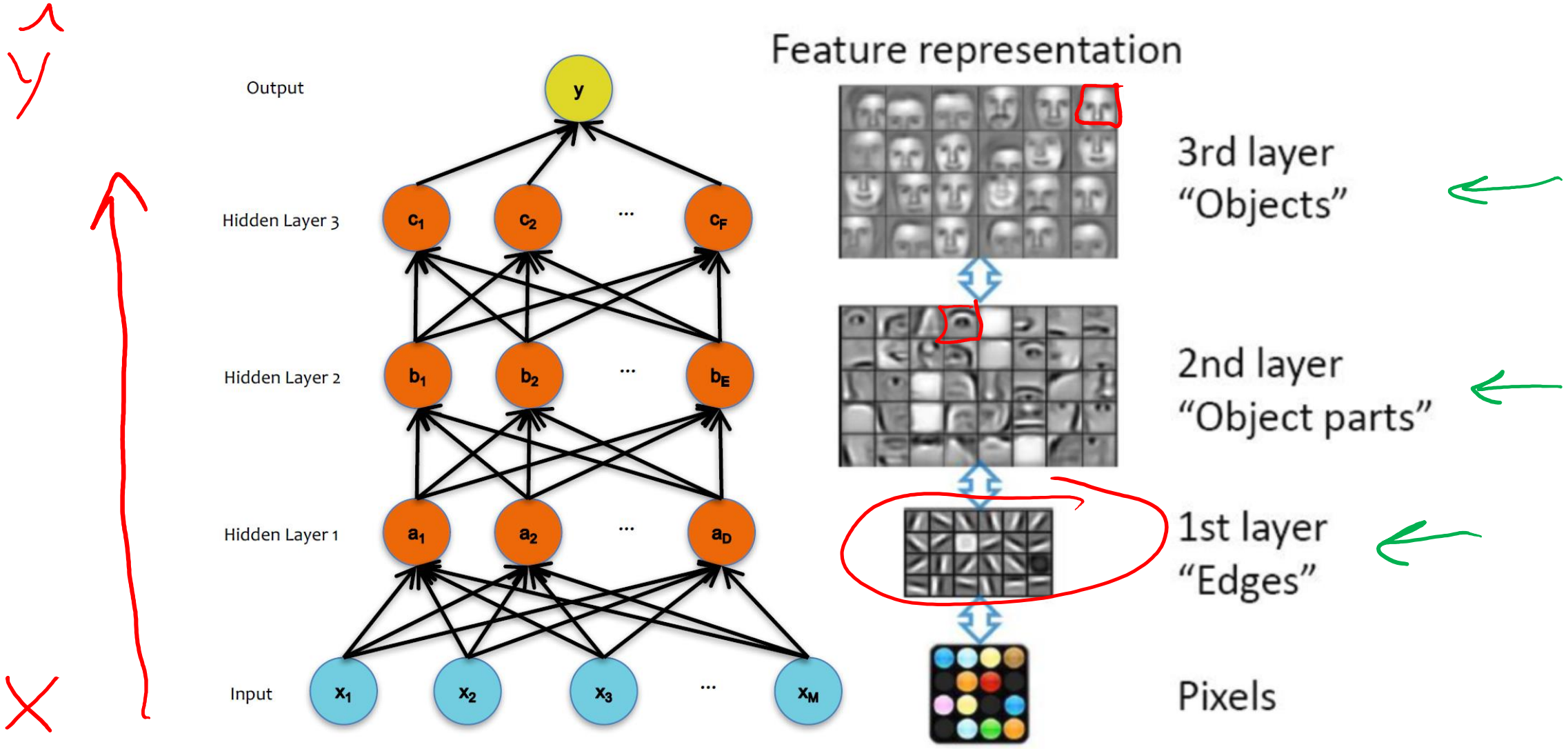
Image Classification

HOG features passed to a linear classifier (SVM)



CAT

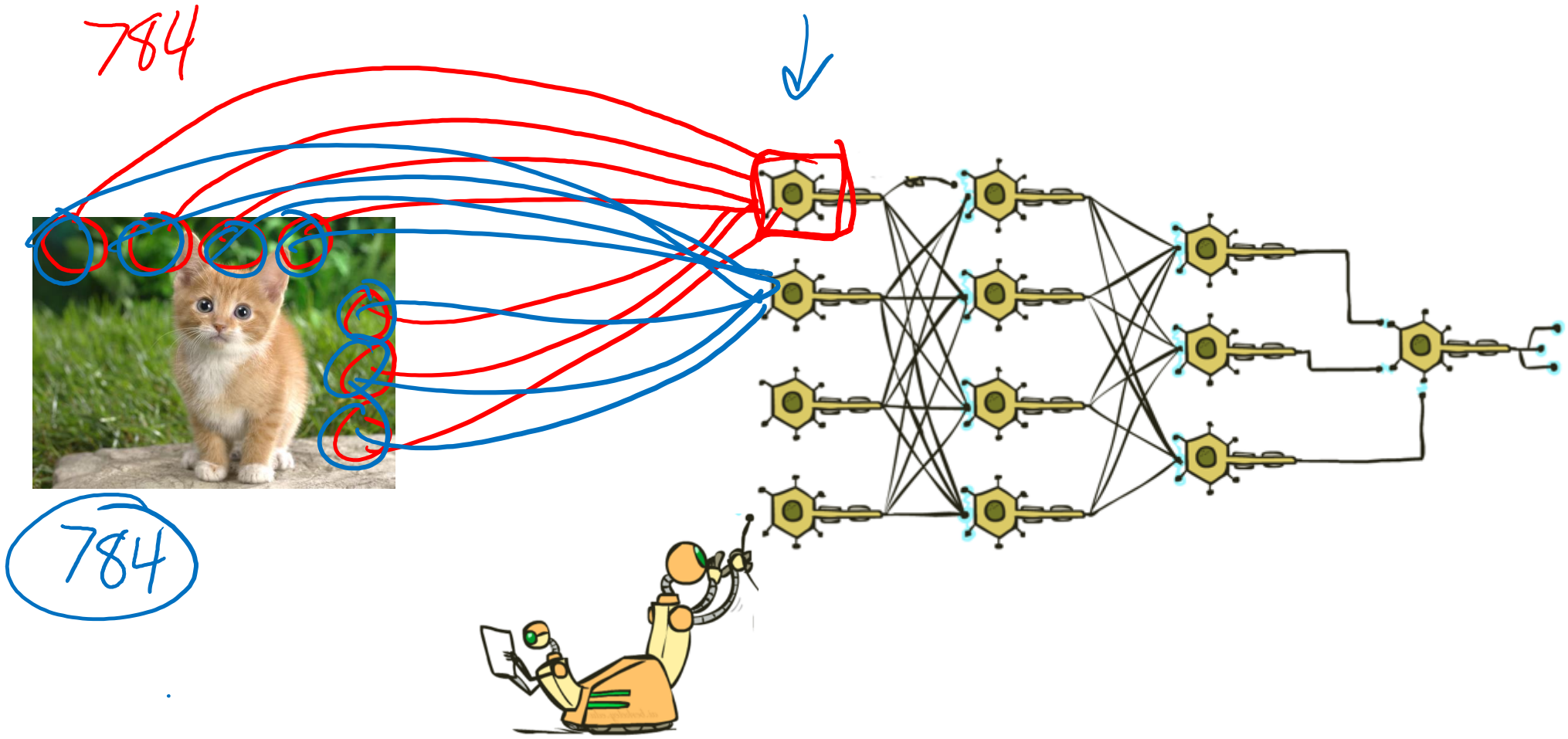
Classification: Learning Features



Example from Honglak Lee (NIPS 2010)

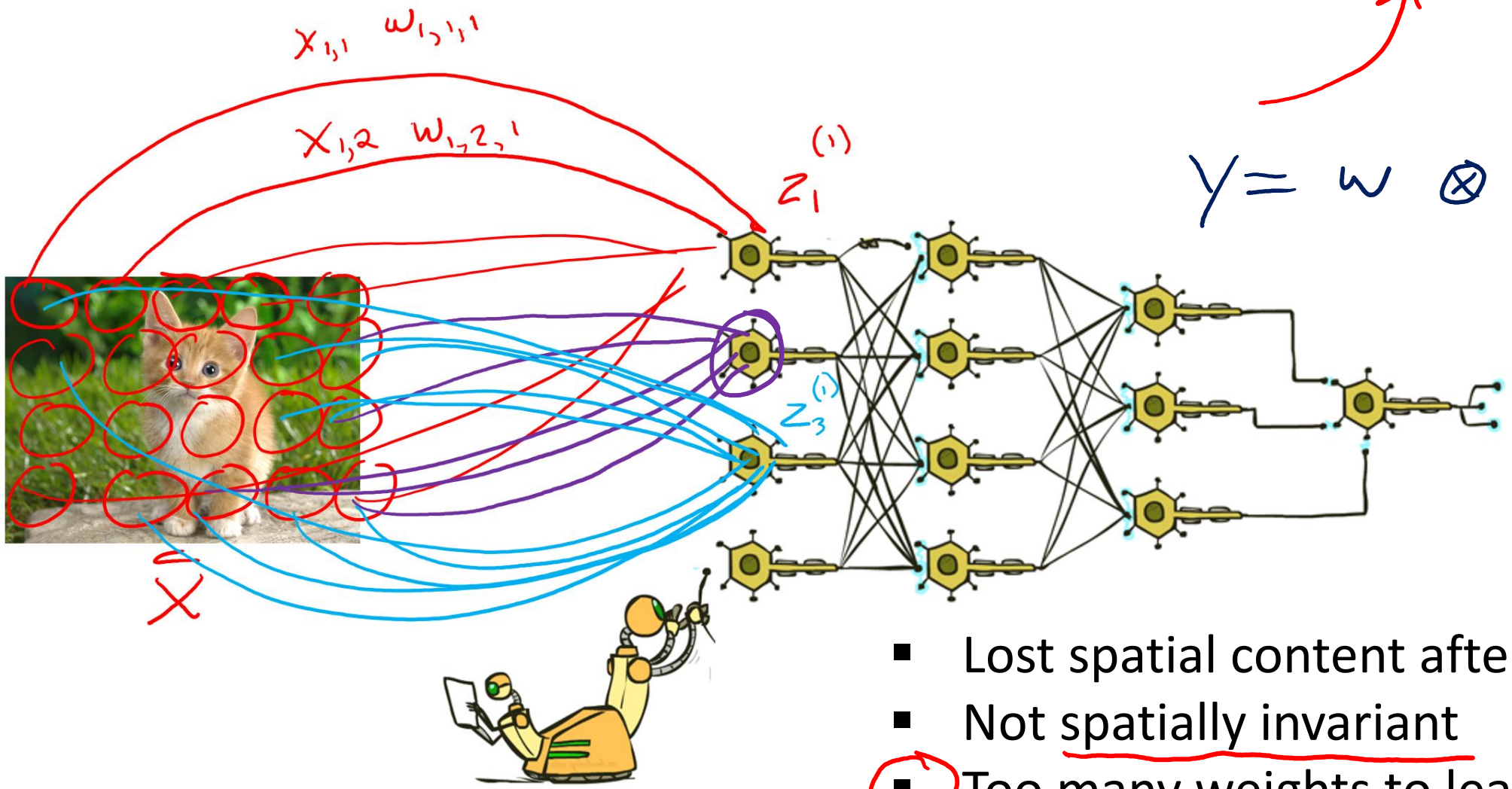
Classification: Deep Learning

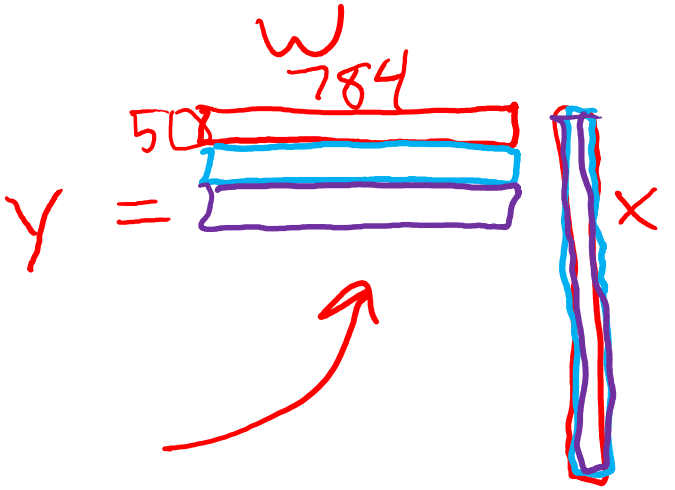
Fully connected neural network?



Classification: Deep Learning

fully connected





$y = w \otimes x$

CAT

- Lost spatial content after first layer
- Not spatially invariant
- Too many weights to learn

Convolution

Signal processing definition

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

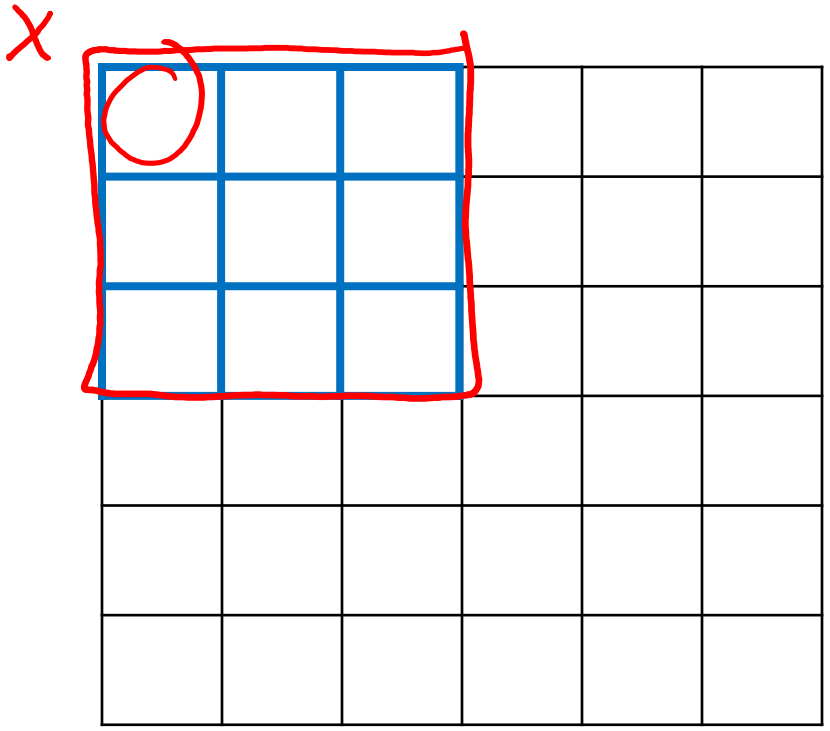
Relaxed definition

- Drop infinity; don't flip kernel

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

w

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



Convolution

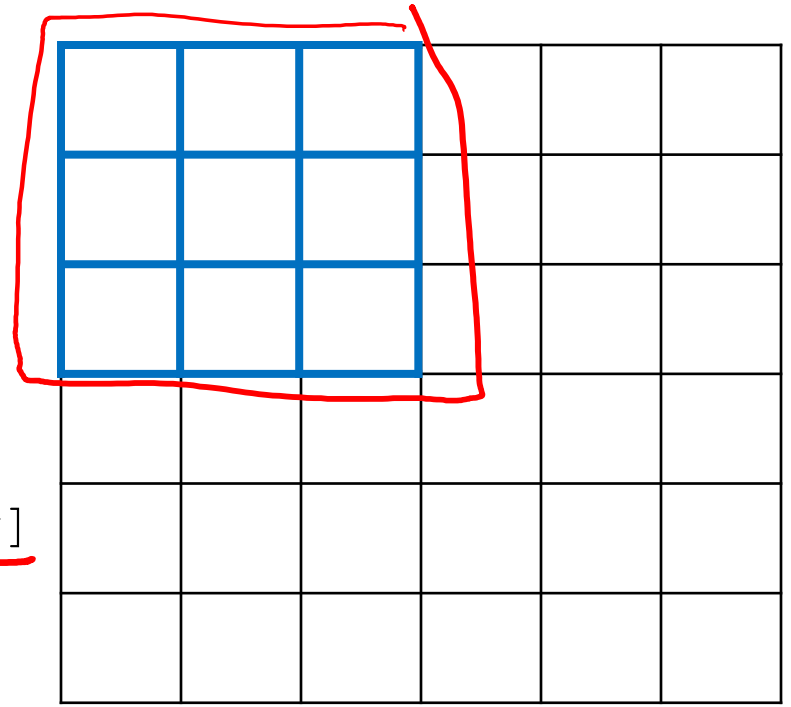
Relaxed definition

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

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

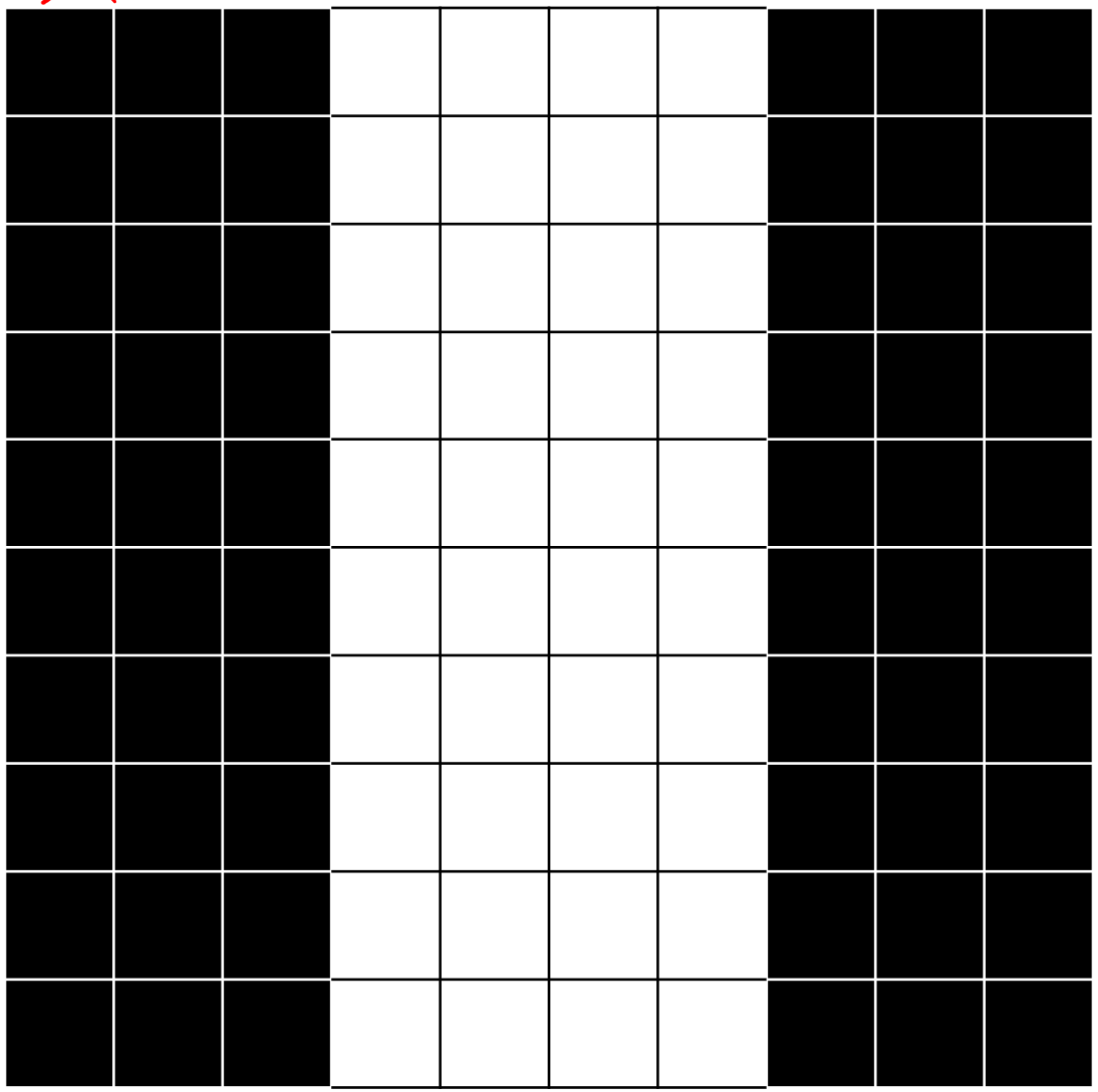
```
for i in range(0, im_width - K + 1):
    for j in range(0, im_height - K):
        im_out[i,j] = 0
        for u in range(0, K):
            for v in range(0, K):
                im_out[i,j] += im[i+u, j+v] * kernel[u,v]
```

GPU!!



Convolution

X



K

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

Convolution

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

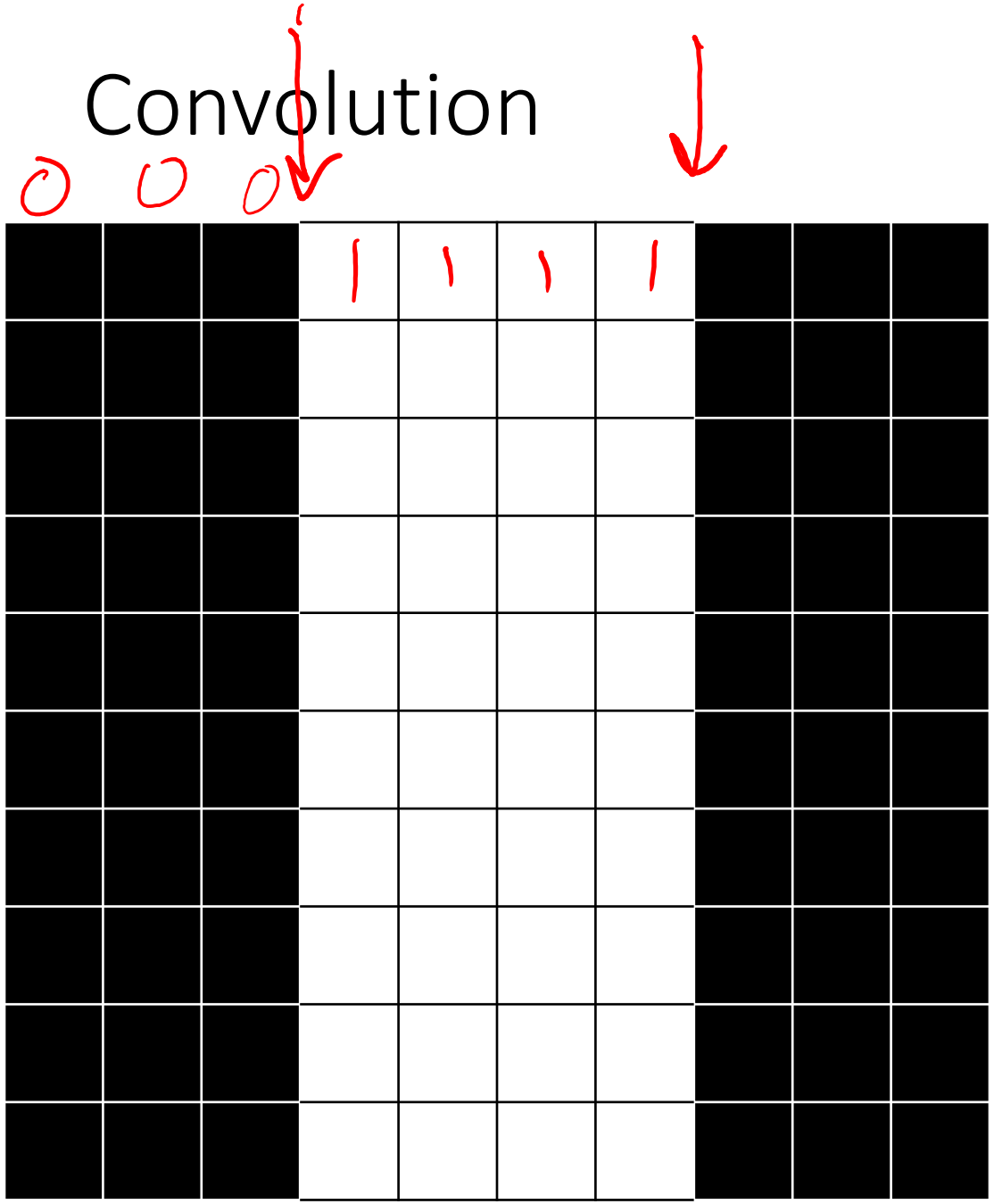
0 3 3 0 0 -3 -3 0

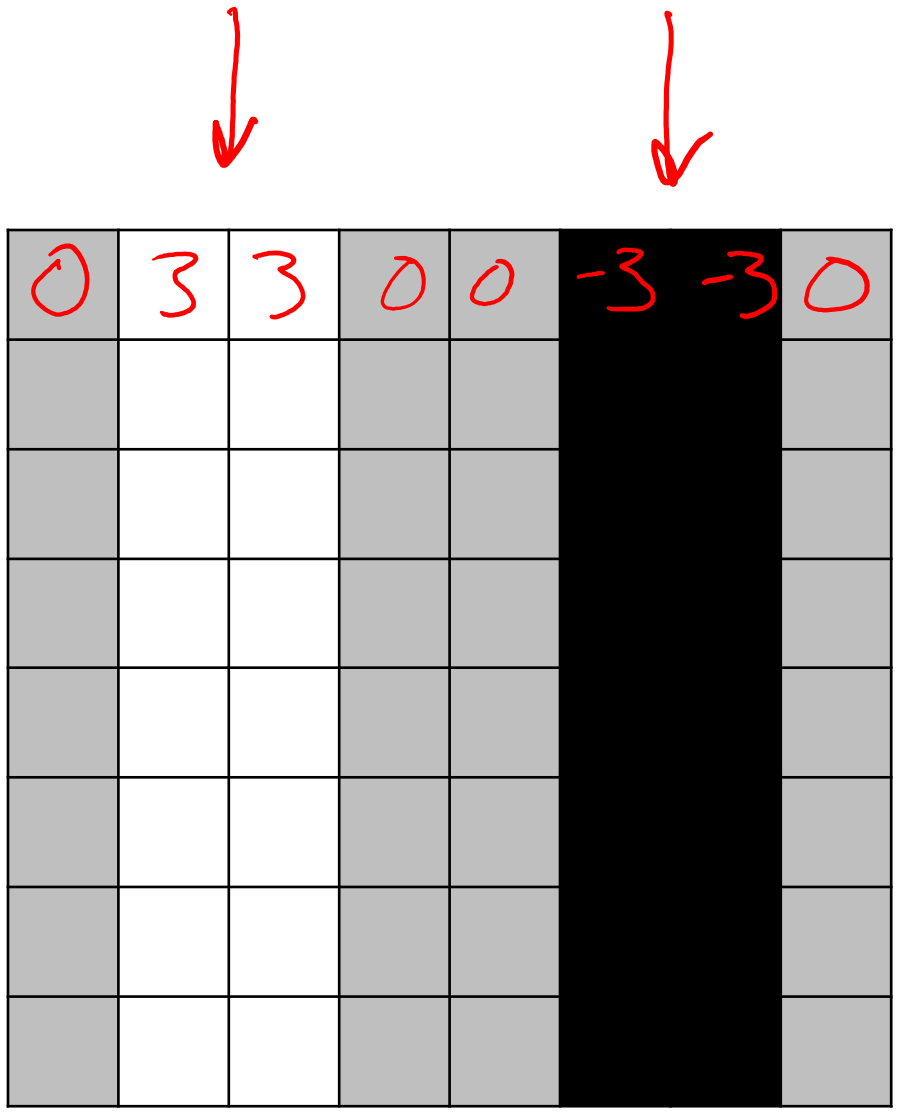
0

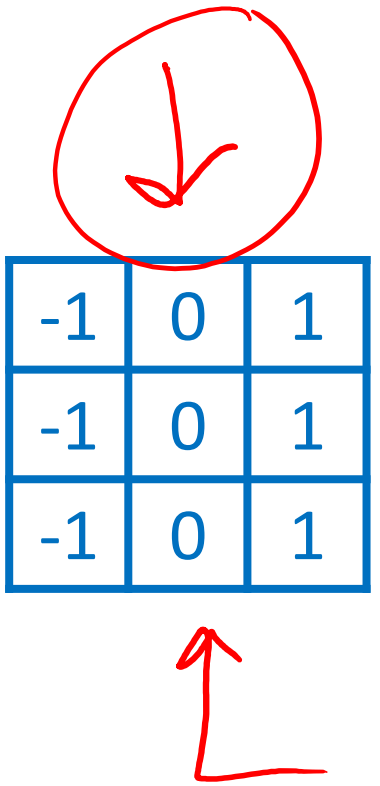
-1	0	1
-1	0	1
-1	0	1

0

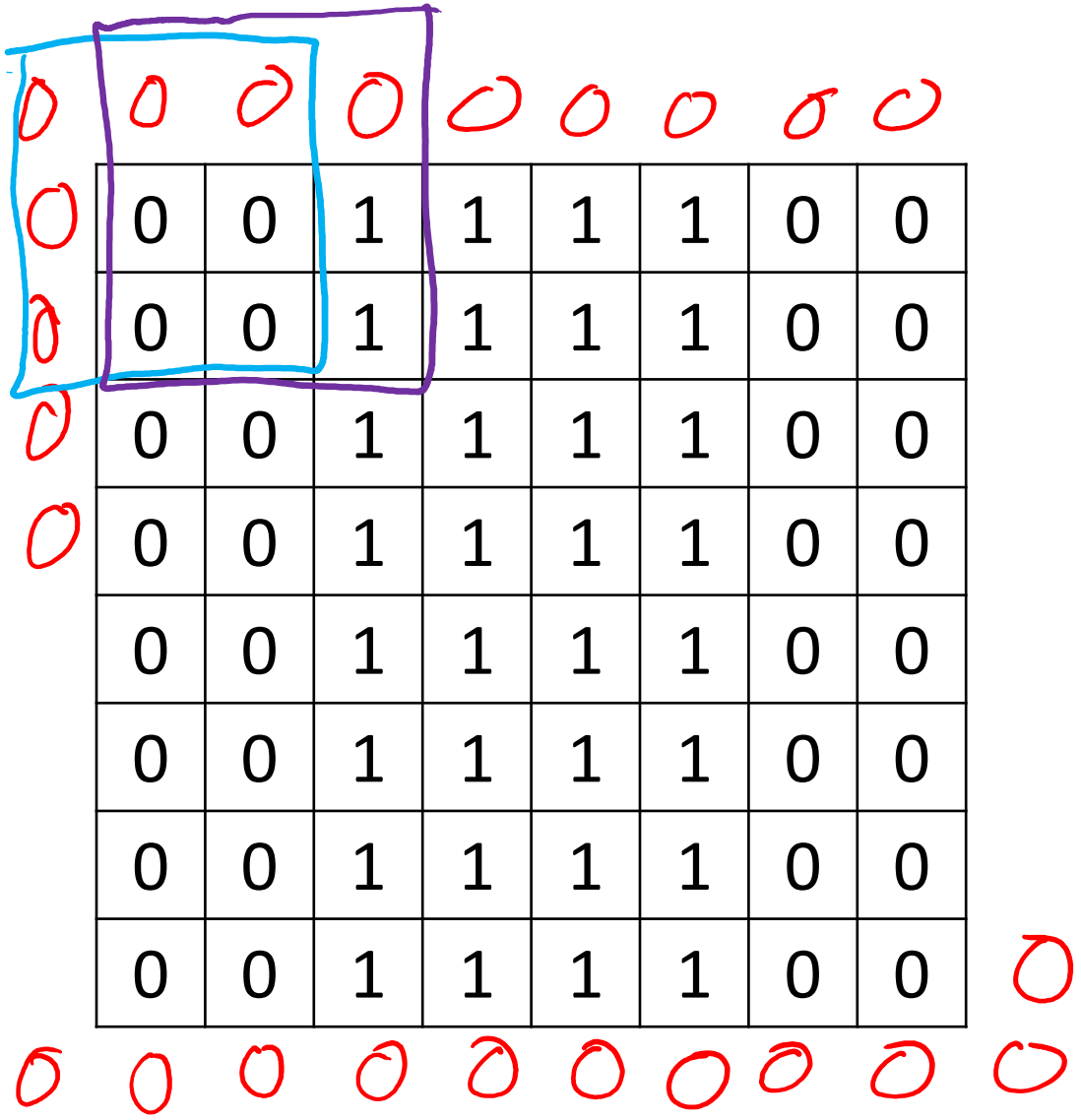
Convolution







Convolution: Padding



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 2 : 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

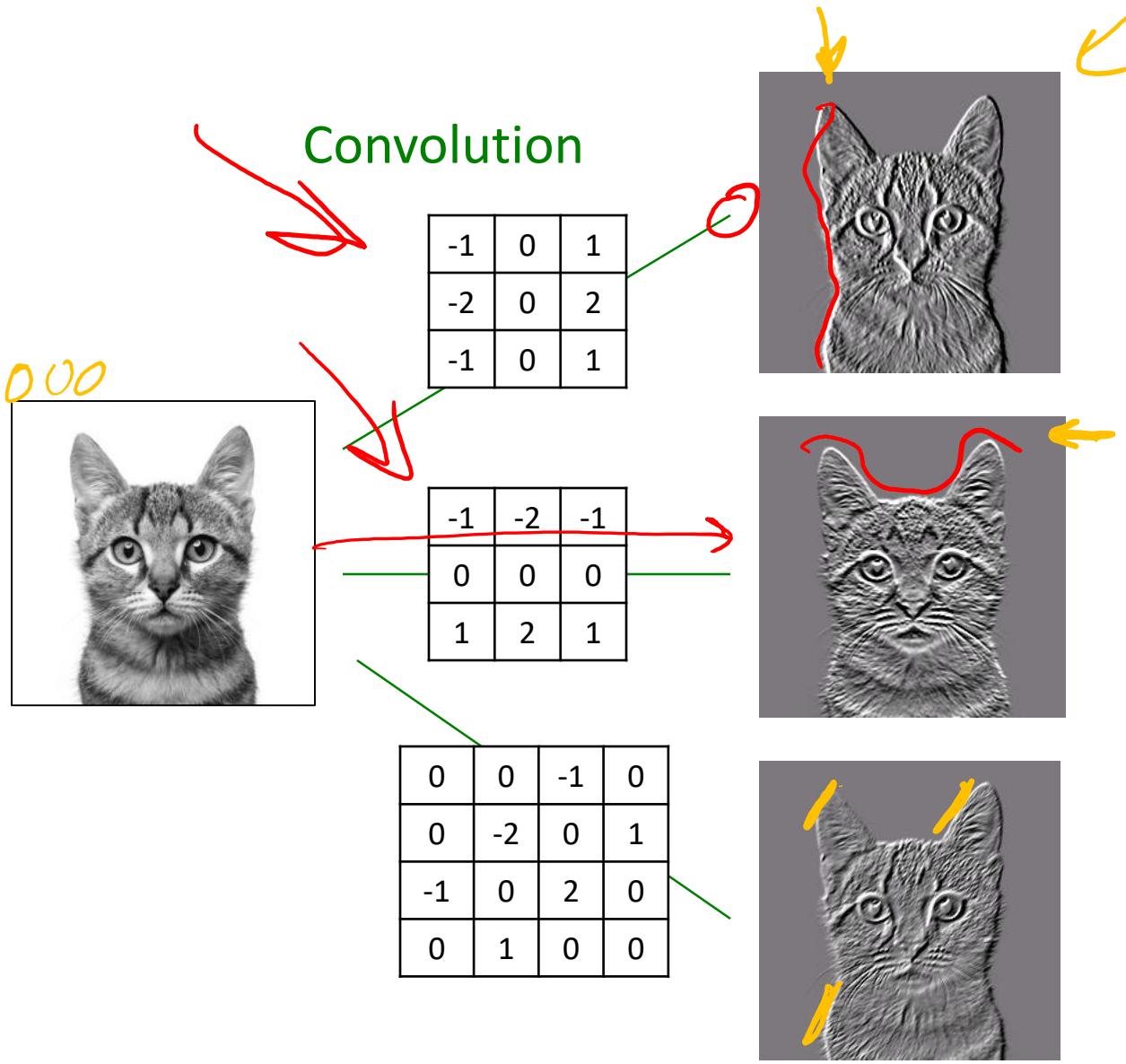
0	0	-1	0
0	-2	0	1
-1	0	2	0
0	1	0	0

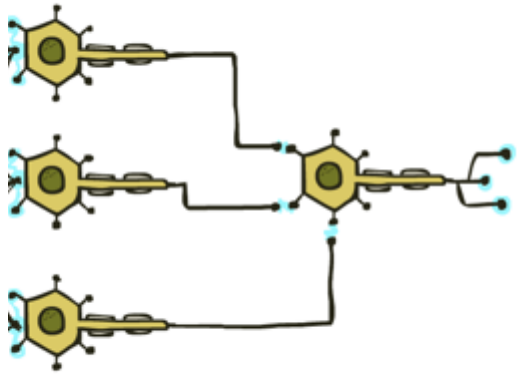
Im1

Im2

Im3

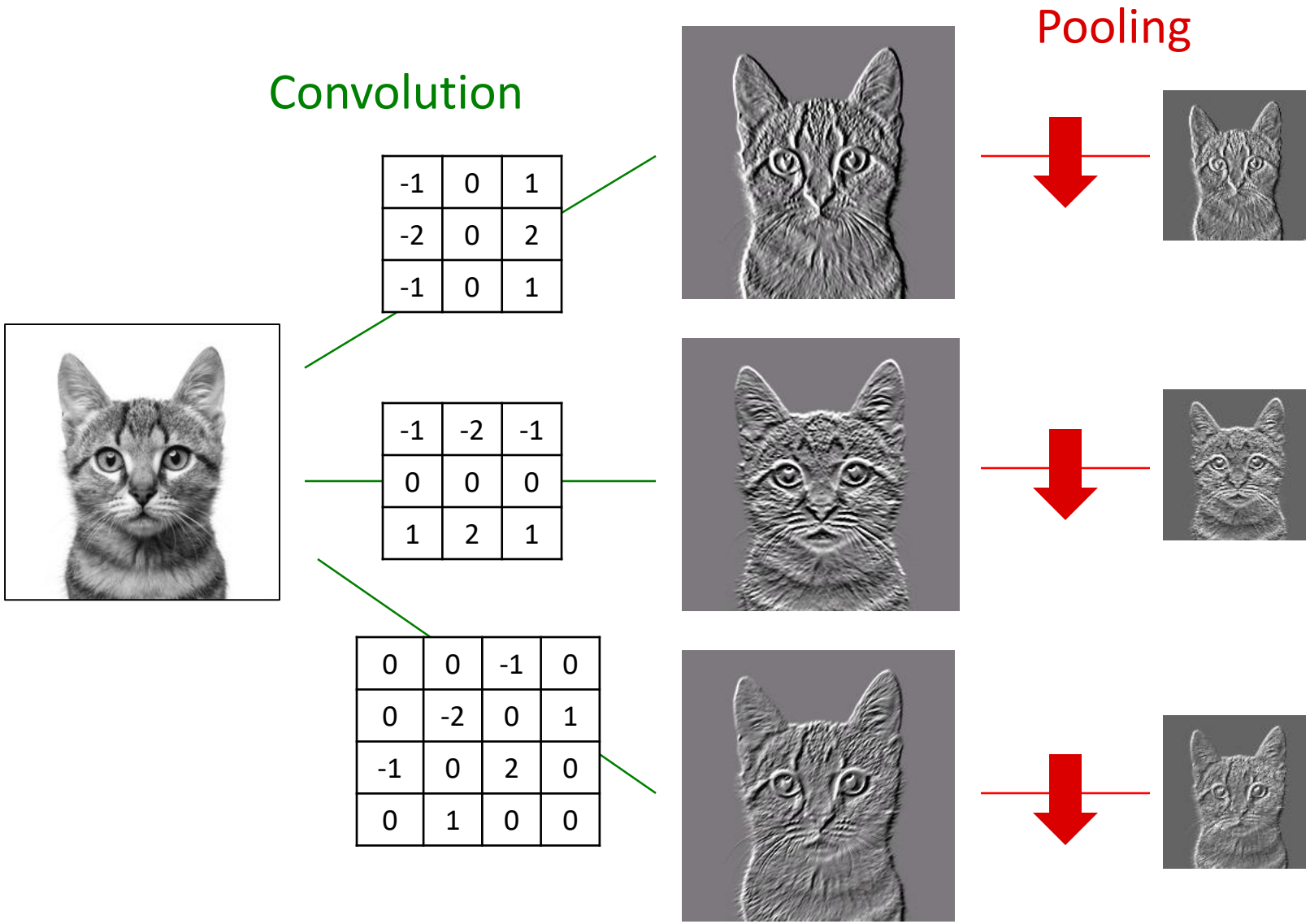
Convolutional Neural Networks





CAT

Convolutional Neural Networks

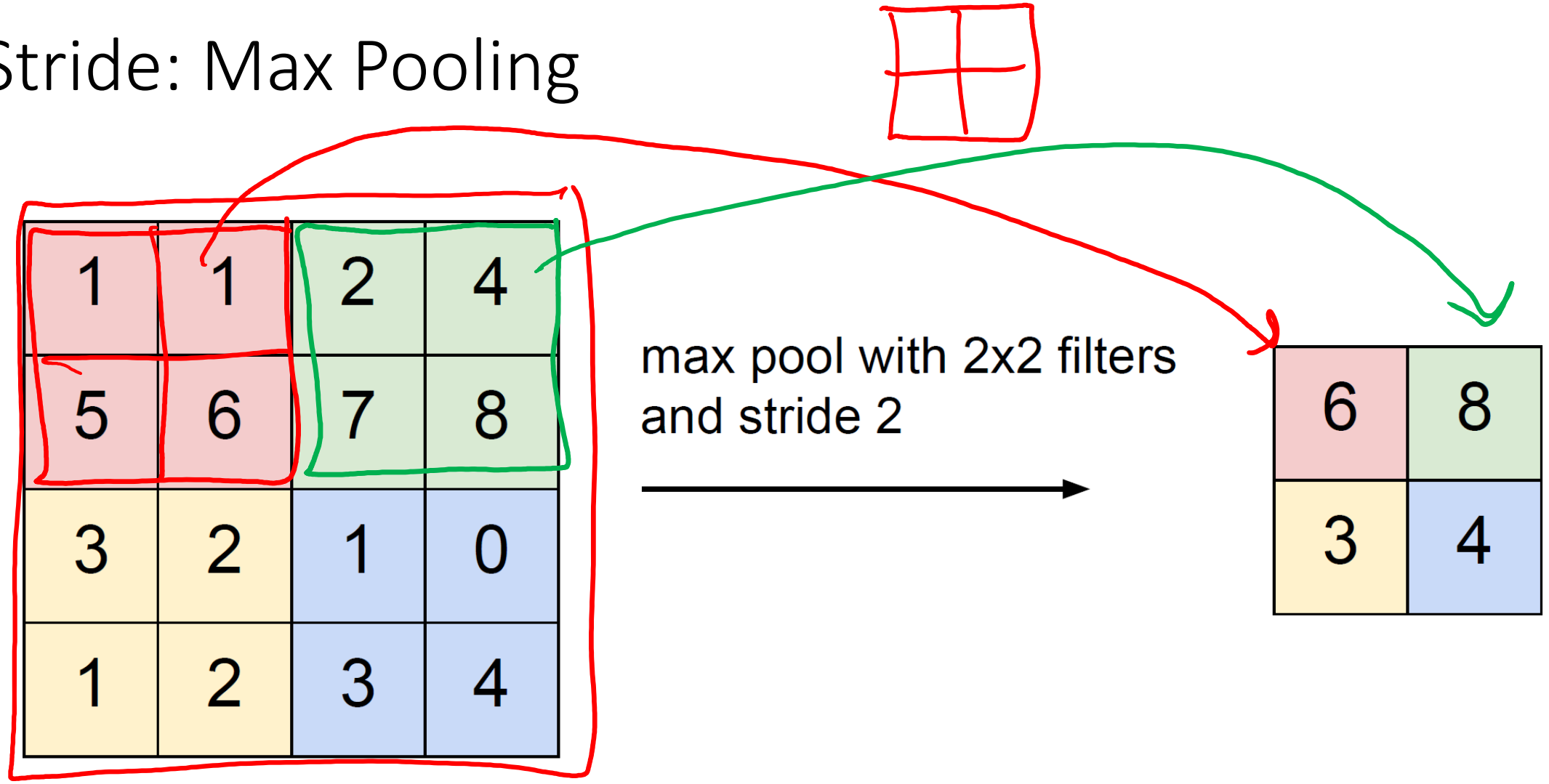


Convolution: Stride=2

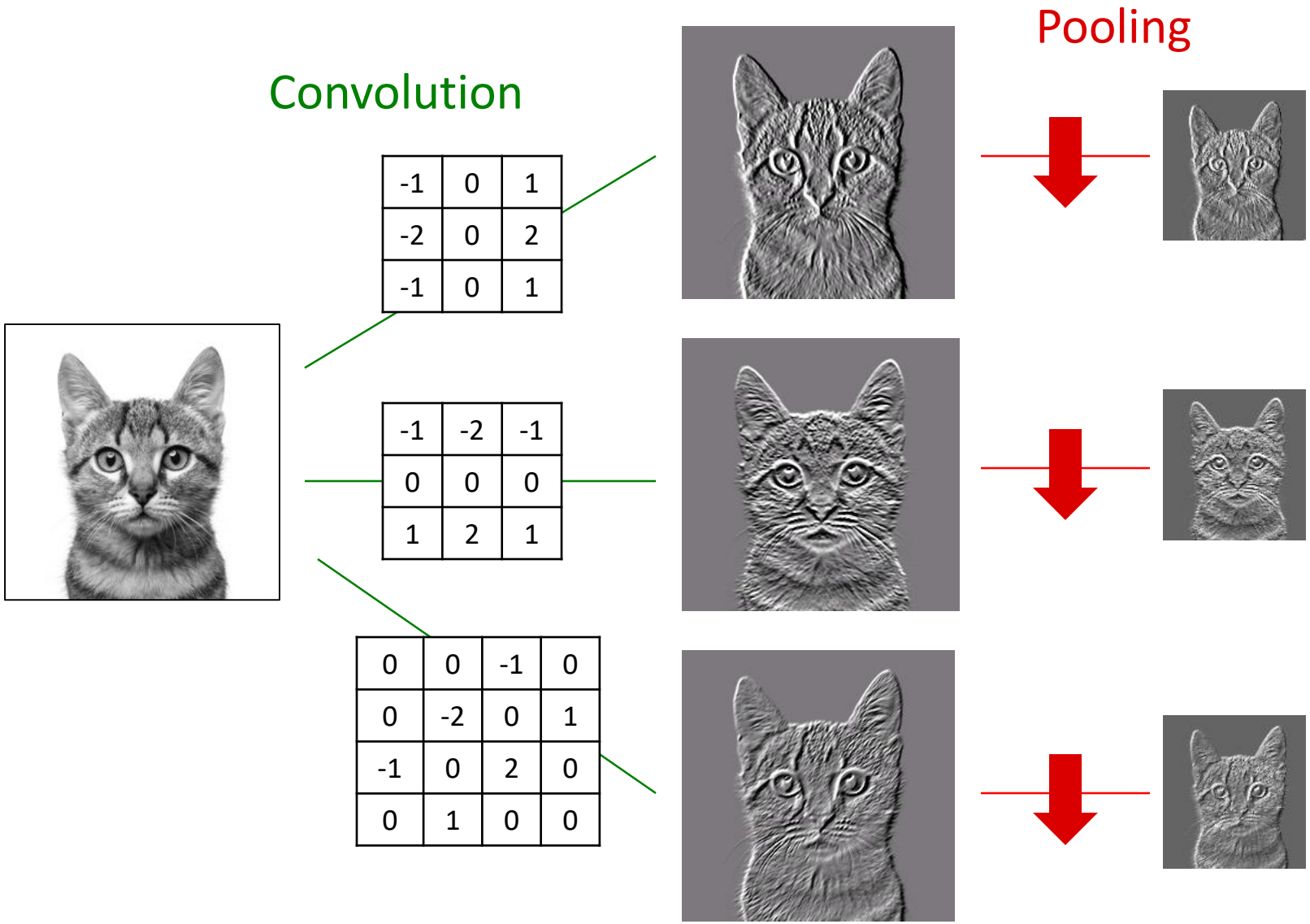
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0

.25	.25
.25	.25

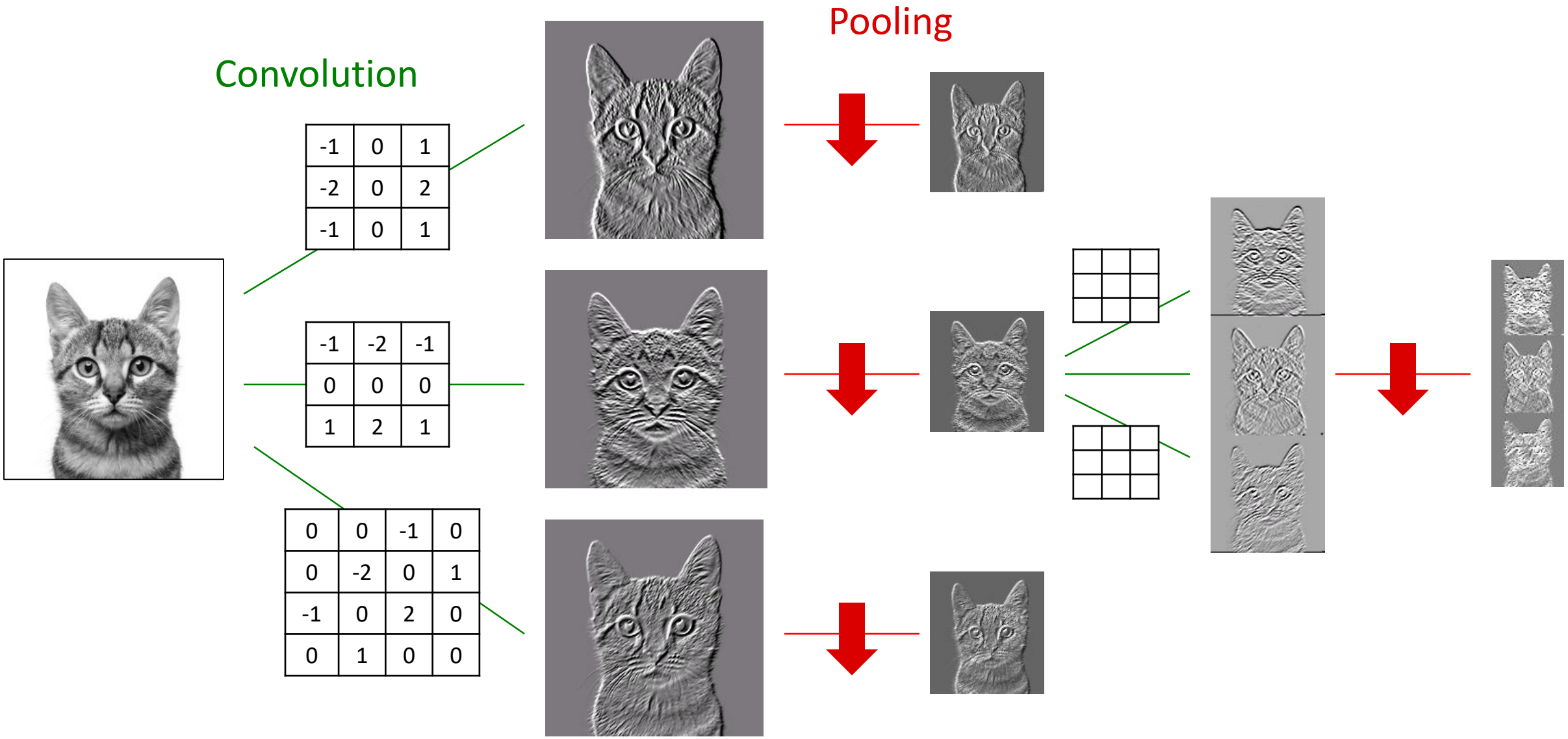
Stride: Max Pooling



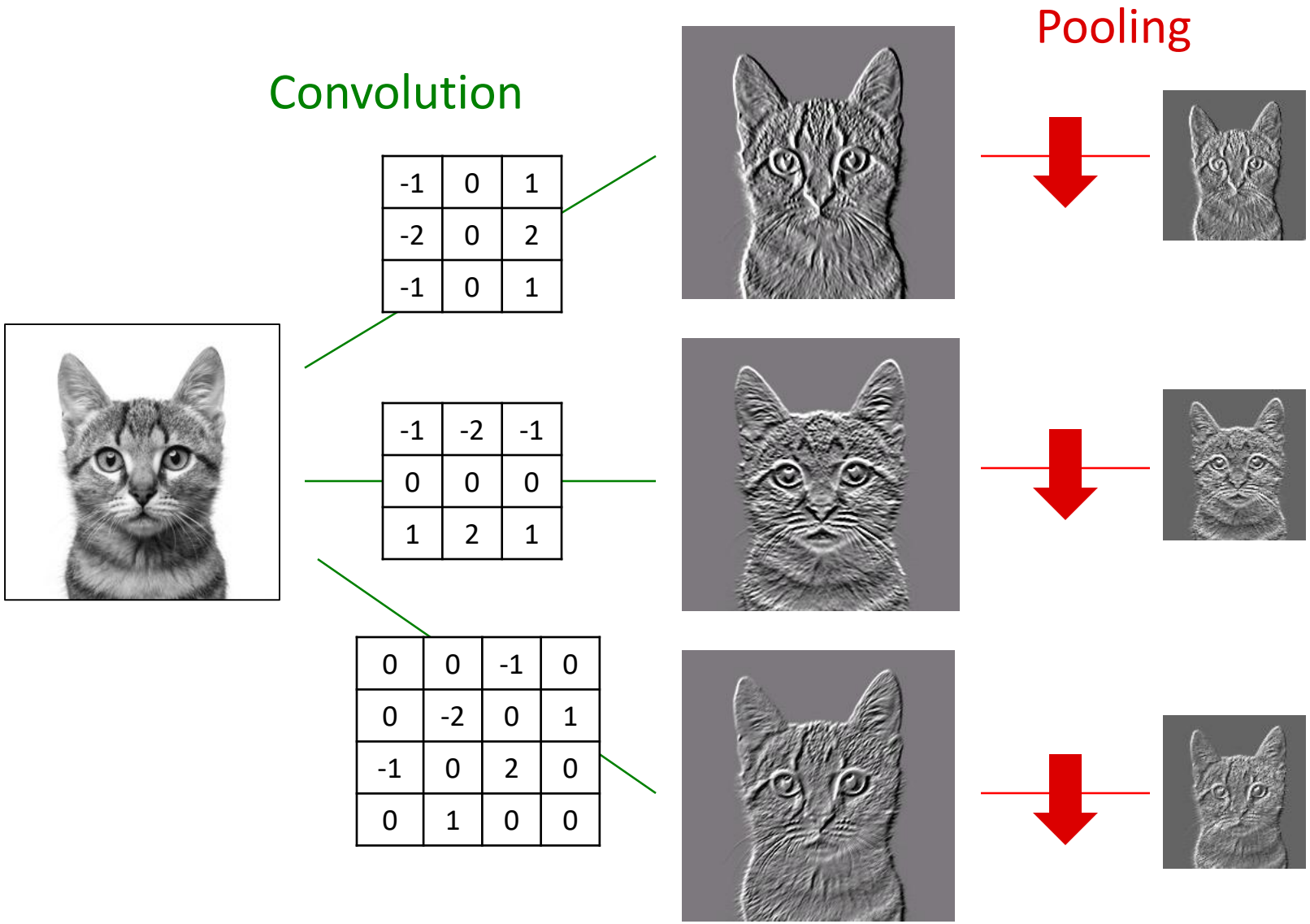
Convolutional Neural Networks



Convolutional Neural Networks



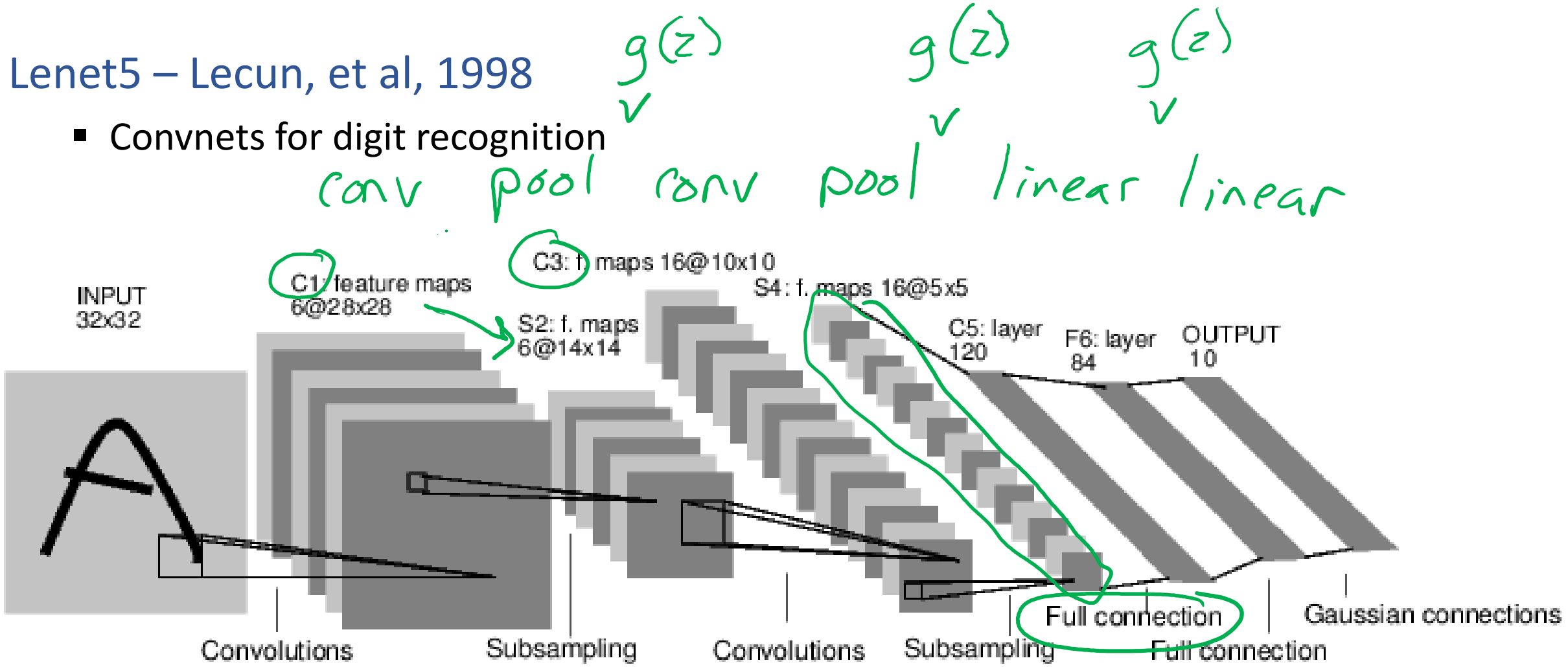
Convolutional Neural Networks



Convolutional Neural Networks

Lenet5 – Lecun, et al, 1998

- Convnets for digit recognition

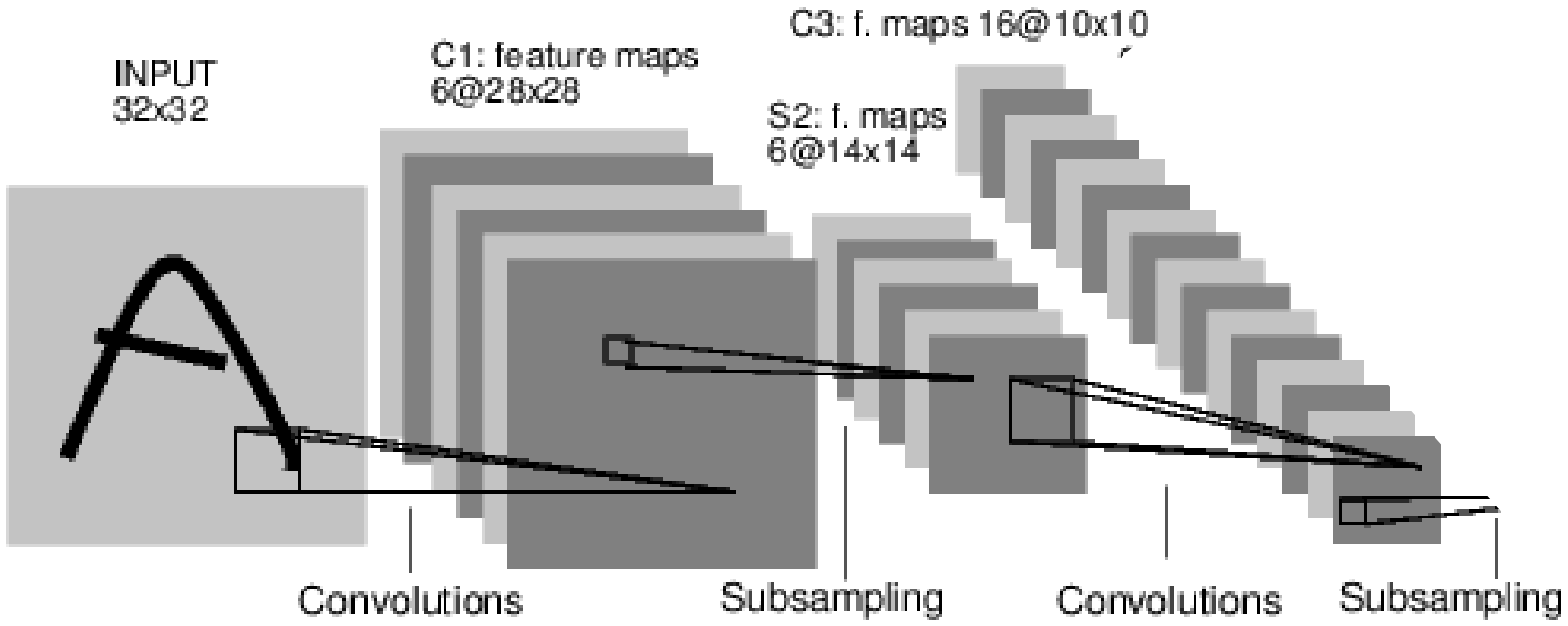


LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

Question:

How big many convolutional weights between S2 and C3?

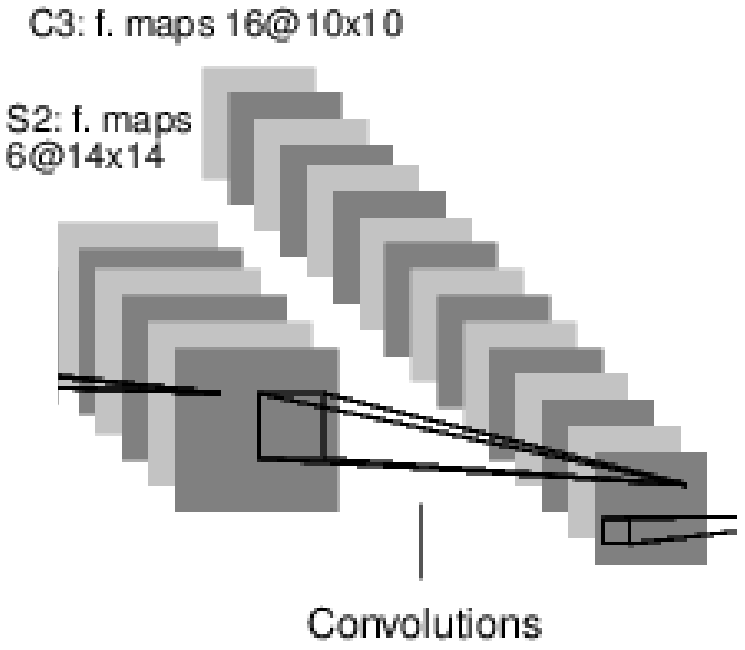
- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10



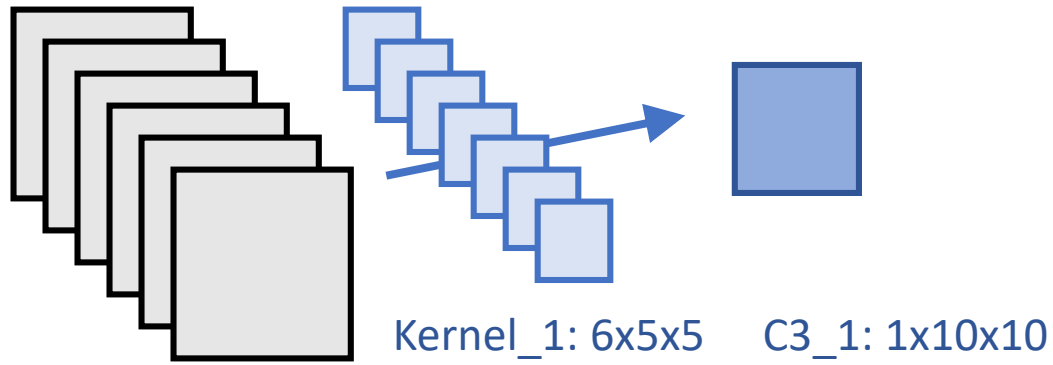
Question:

How big many convolutional weights between S2 and C3?

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



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



S2: 6x14x14

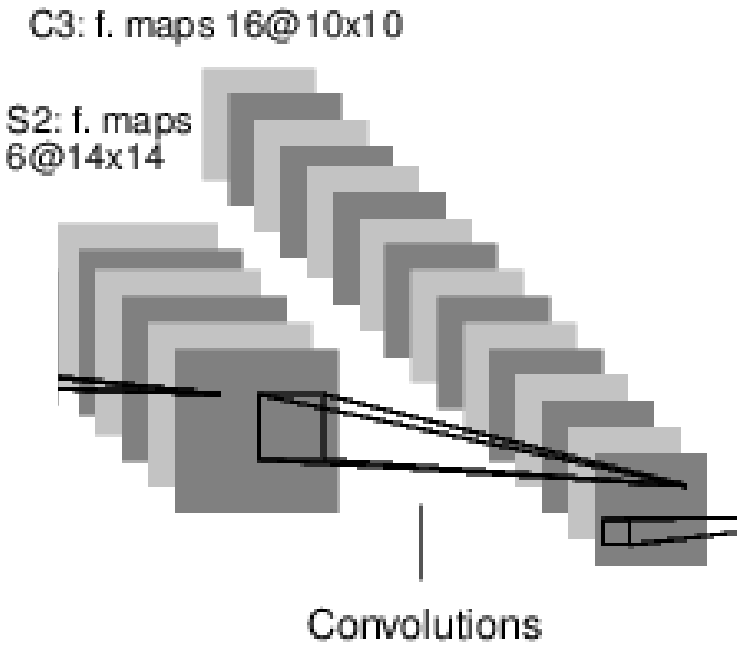
Kernel_1: 6x5x5

C3_1: 1x10x10

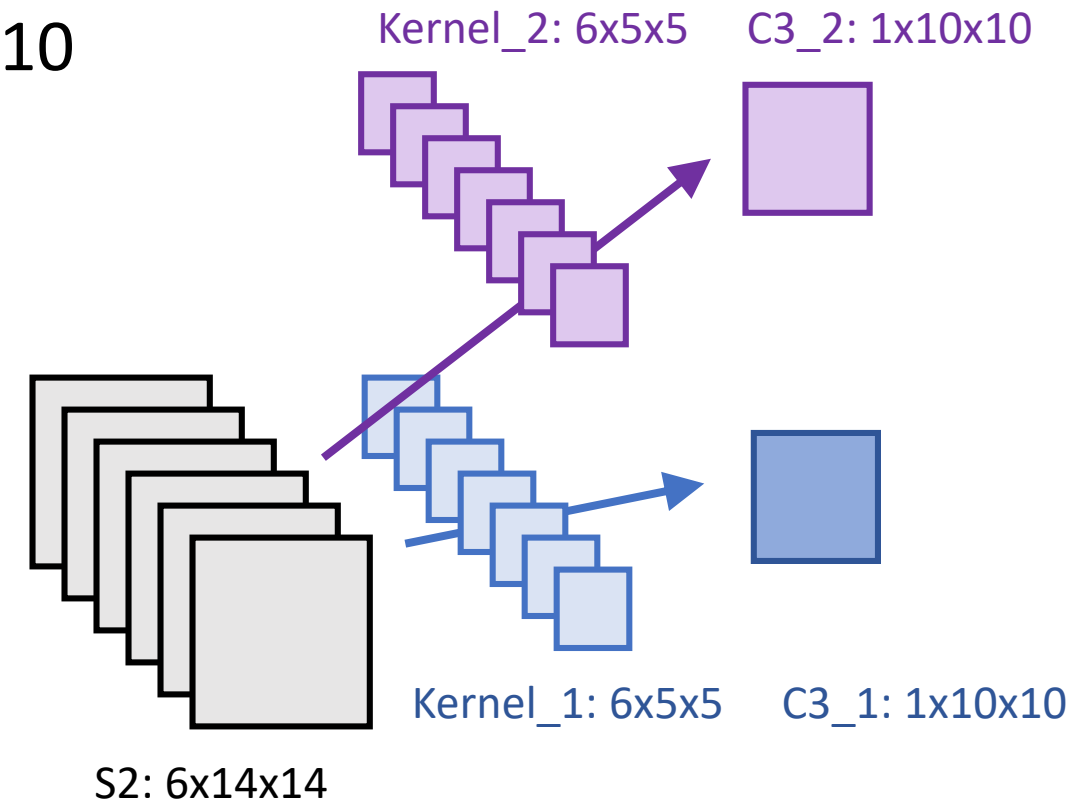
Question:

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

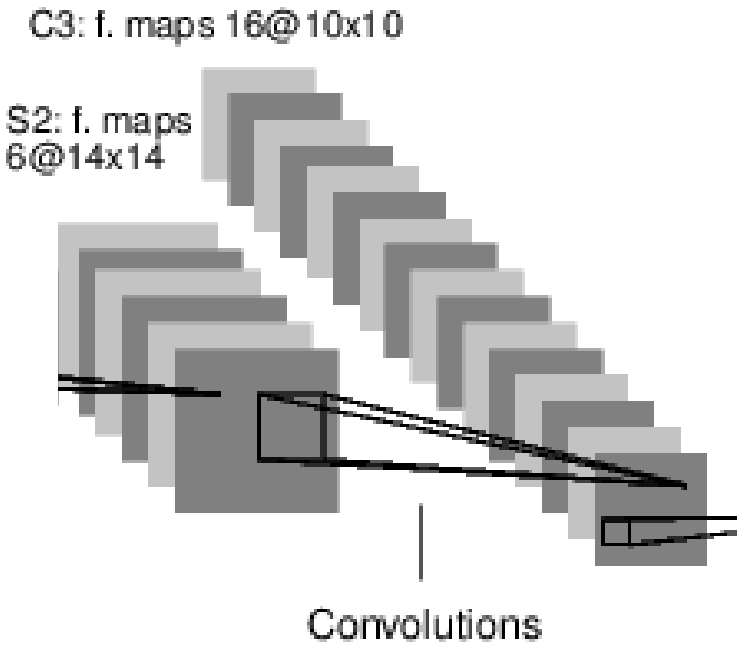


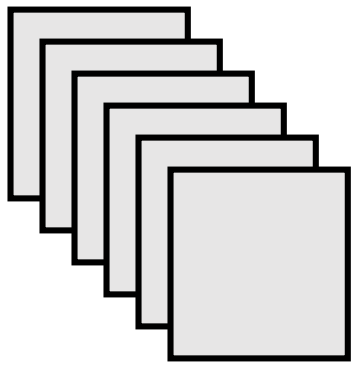
Question:

How big many convolutional weights between S2 and C3?

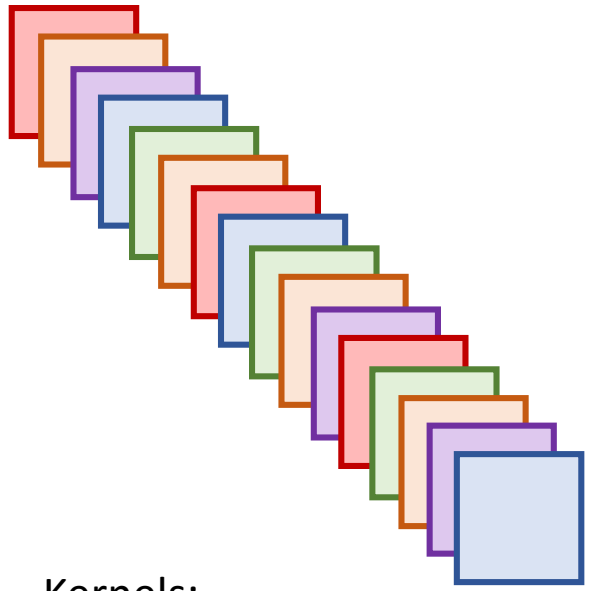
- S2: 6 channels @14x14
- Conv: 5x5, pad=0, stride=1
- C3: 16 channels @ 10x10

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!





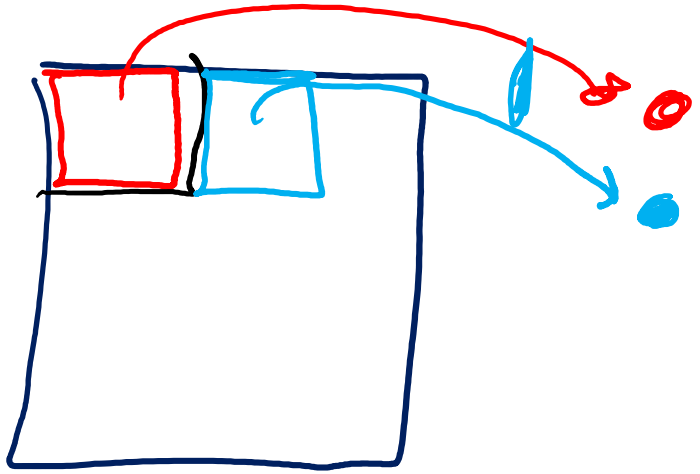
S2: 6x14x14



Kernels:
16@6x5x5

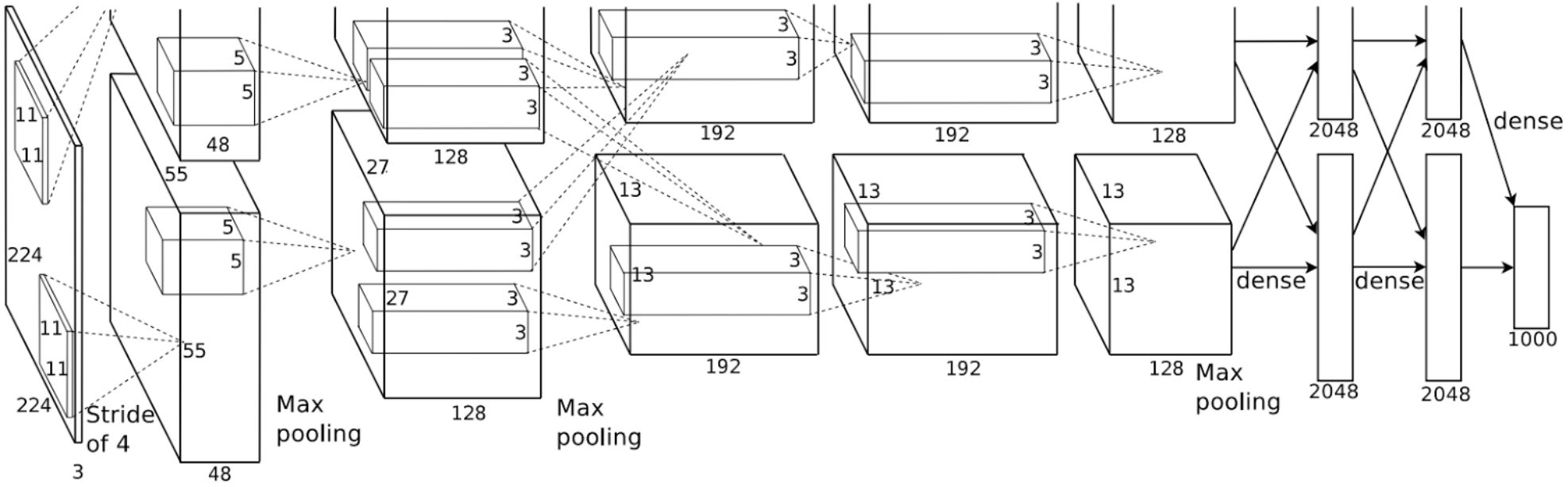
C3: 16@10x10

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.