Convolutional networks

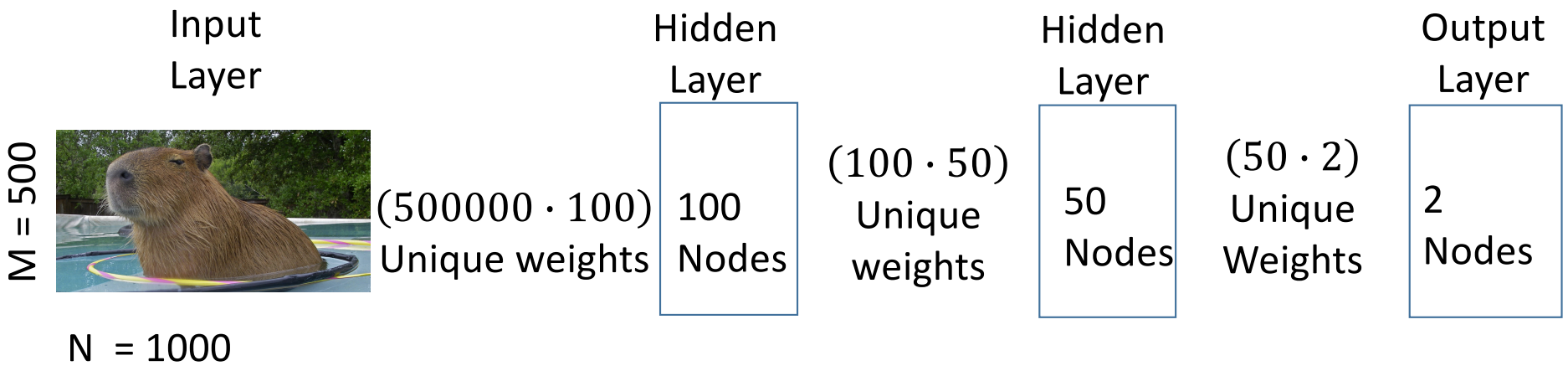
Deep Learning: Bryan Pardo, Northwestern University, Fall 2020

How big is that image?

500

1000

How many weights in a fully connected net?



$$50{,}000{,}000 + 5{,}000 + 100 = 50{,}005{,}100 \text{ weights}$$

How does the eye do this?

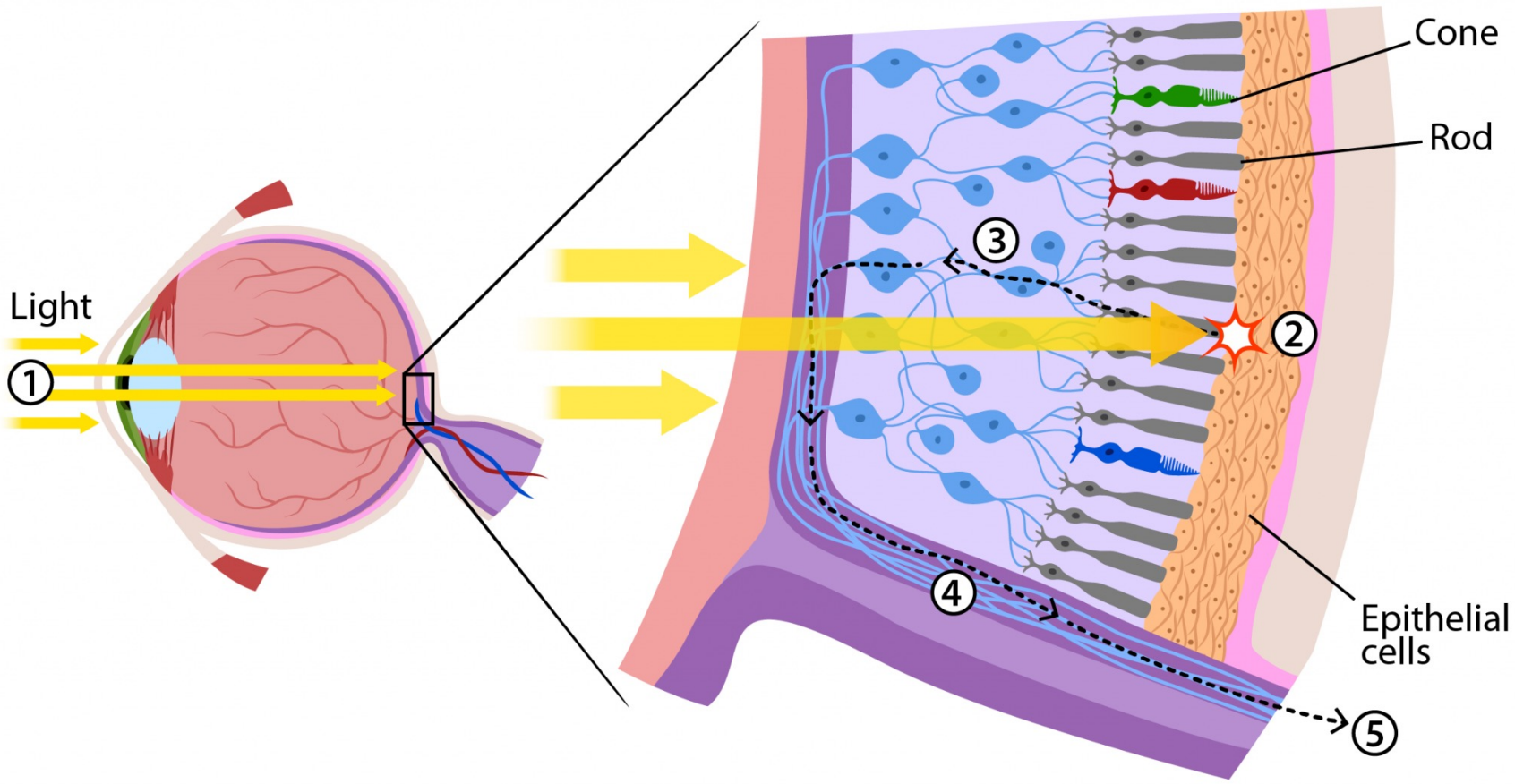


Image from https://askabiologist.asu.edu/rods-and-cones

Limited receptive fields, at multiple levels

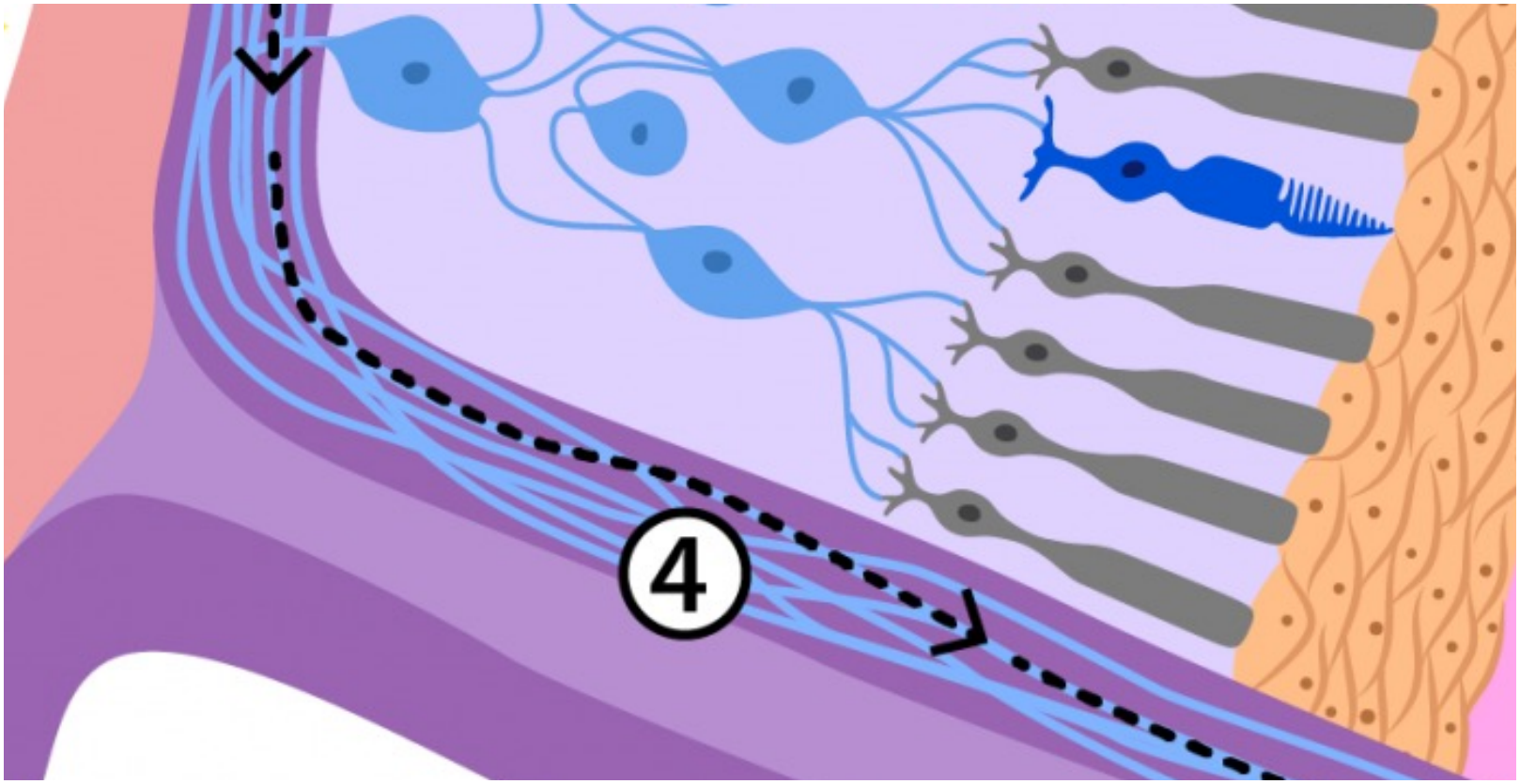
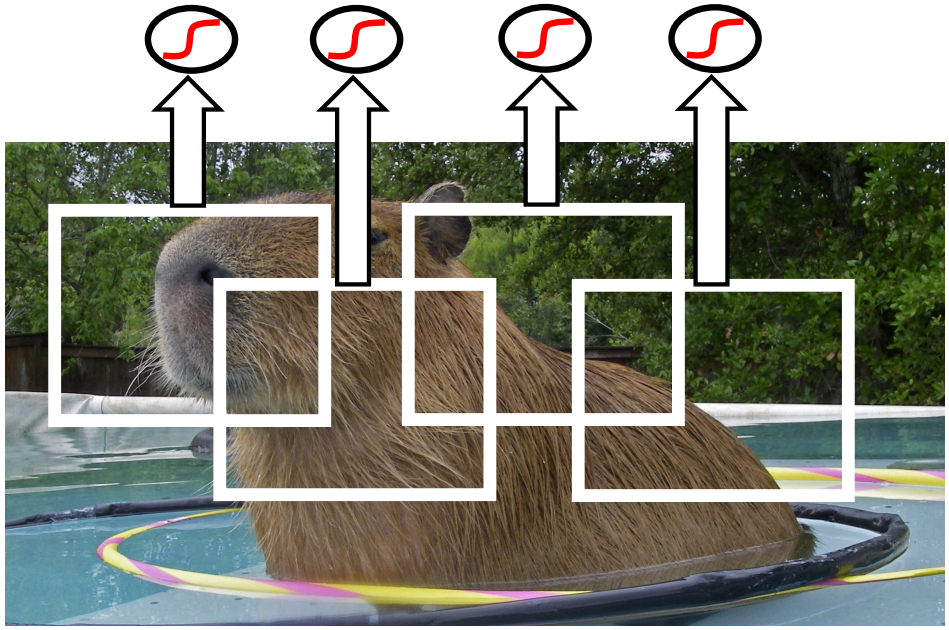


Image from https://askabiologist.asu.edu/rods-and-cones

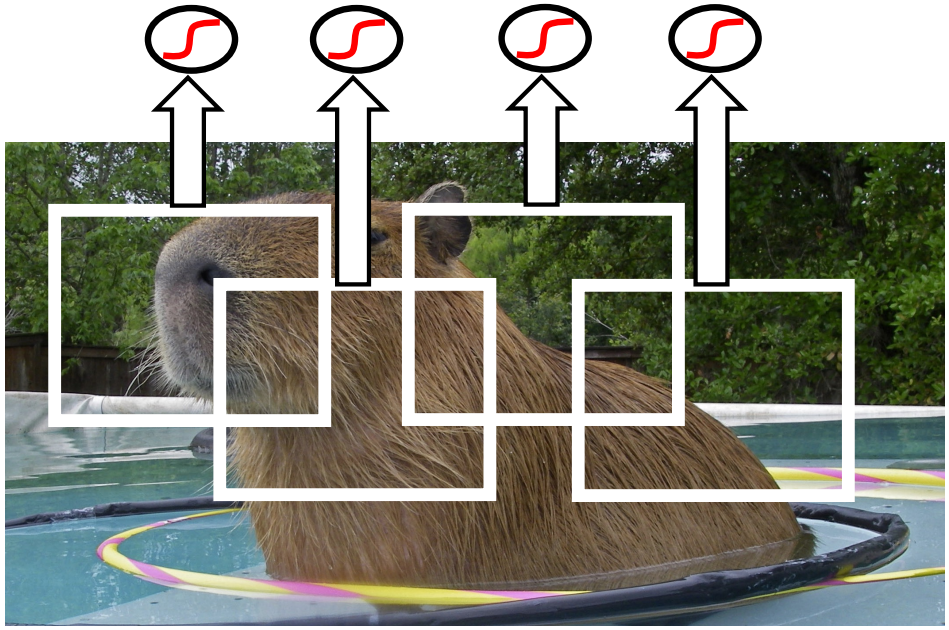
Small Fixed Windows (filter size/receptive field)

- If important features fall within a bounded size region, we can bound the receptive field of each unit to that size.
- This greatly reduces the number of weights.

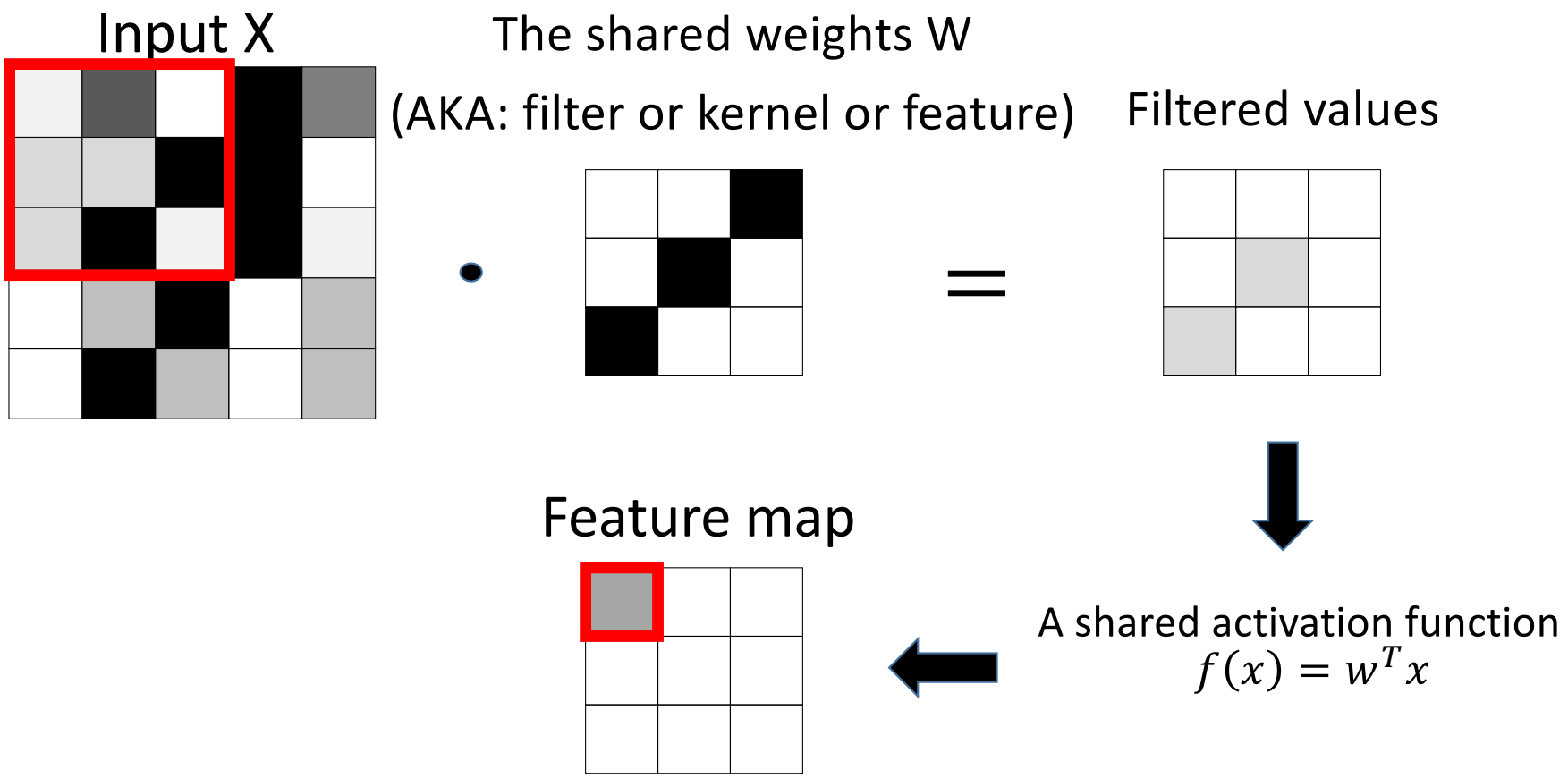


Shared weights

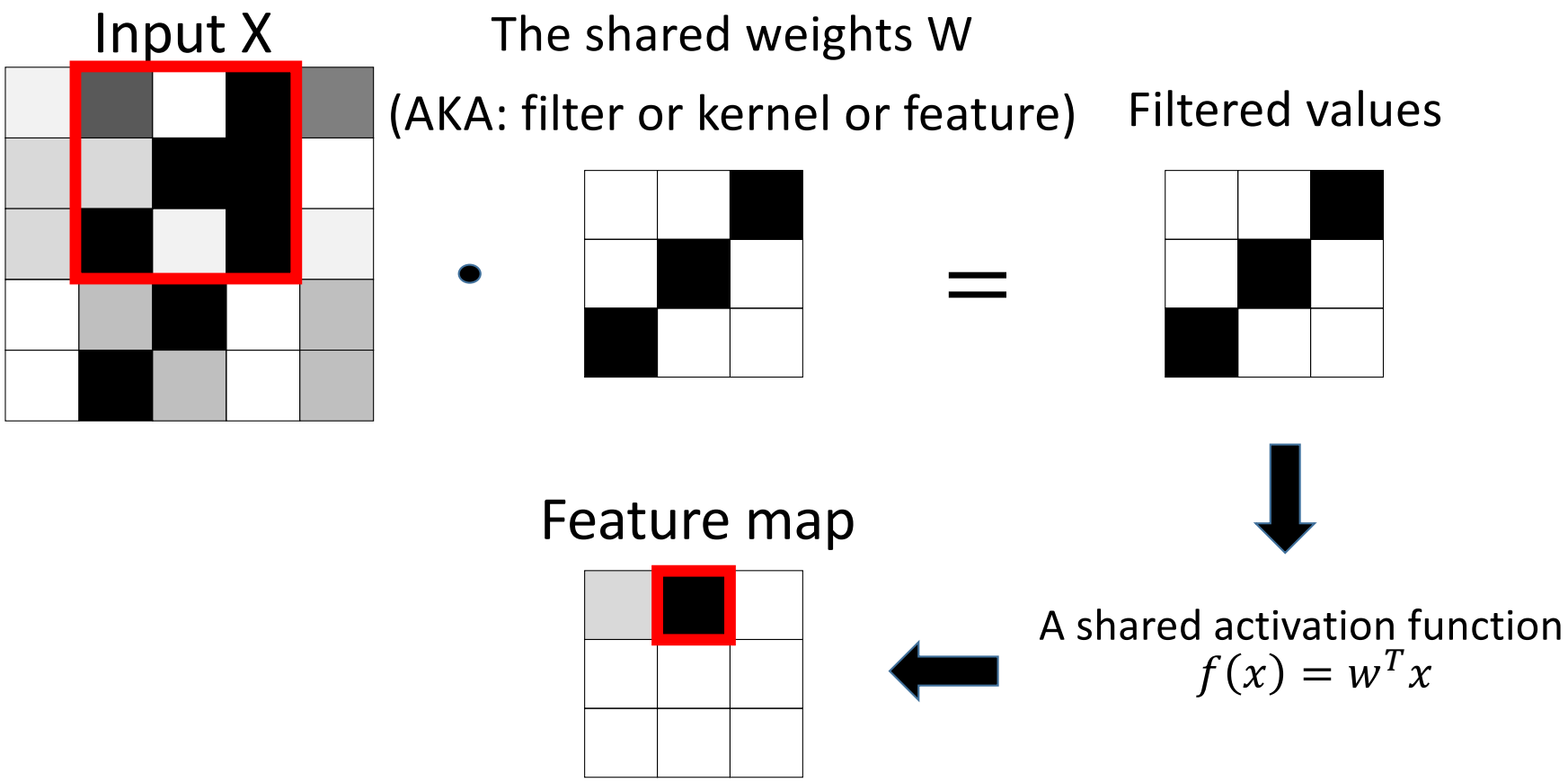
- If a feature is good to find in one region, it may be good to find in other regions.
- Units look for the same feature if they share weights.
- A set of units that share weights is a feature map (aka ”channel”)



Building that feature map

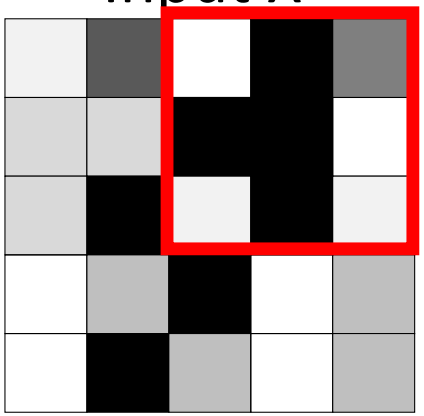


Building that feature map



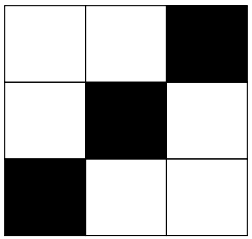
Building that feature map

Input X

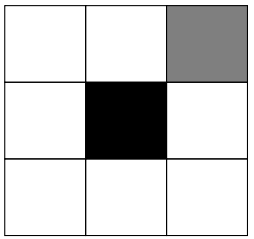


The shared weights W

(AKA: filter or kernel or feature)



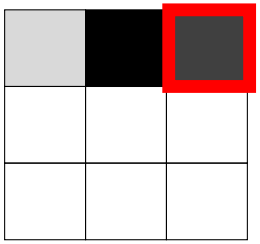
Filtered values



A shared activation function

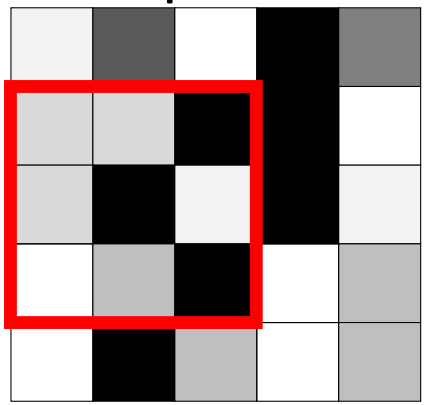
$f(x) = w^T x$

Feature map



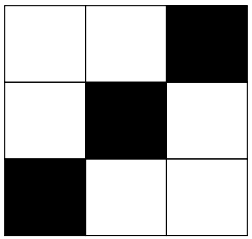
Building that feature map

Input X

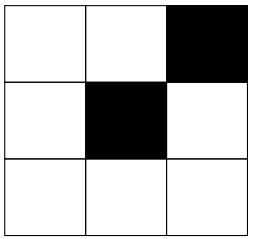


The shared weights W

(AKA: filter or kernel or feature)



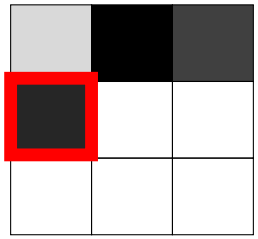
Filtered values



A shared activation function

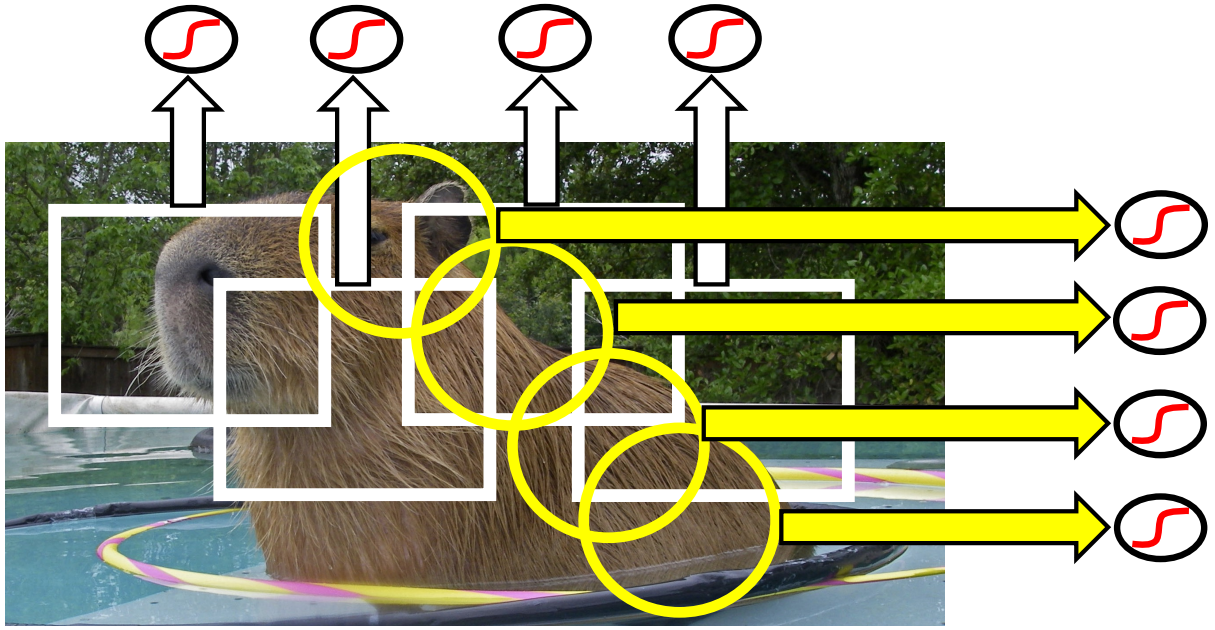
$f(x) = w^T x$

Feature map



Multiple Feature Maps

- To look for multiple features, use multiple feature maps.
- Each map will specialize on one thing.
- Even with many feature maps, you still have far fewer weights

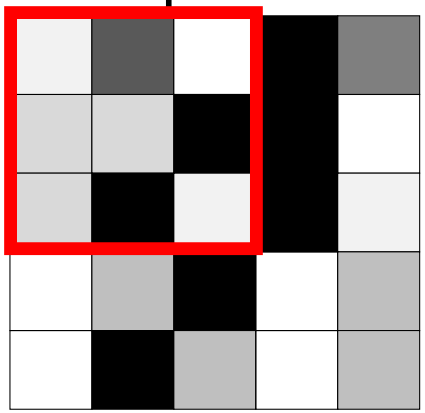


Stride

- How many units you move with each step of your filter/kernel

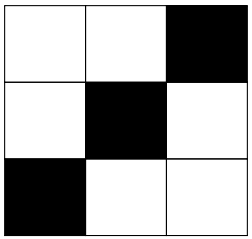
Let's make that stride = 2

Input X

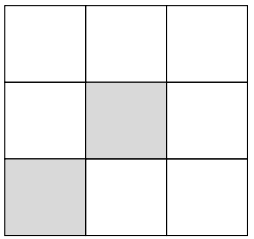


The shared weights W

(AKA a filter, AKA a feature)



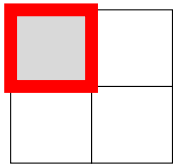
Filtered values



A shared activation function

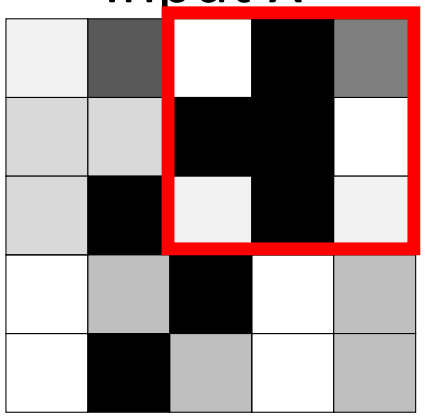
$f(x) = w^T x$

Feature map



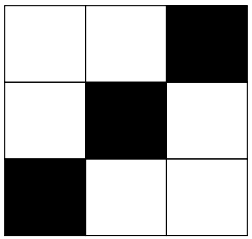
Let's make that stride = 2

Input X

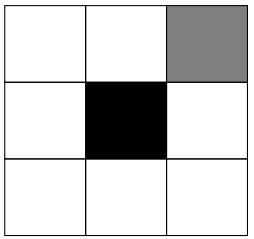


The shared weights W

(AKA a filter, AKA a feature)



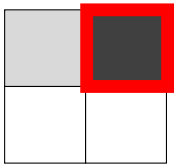
Filtered values



A shared activation function

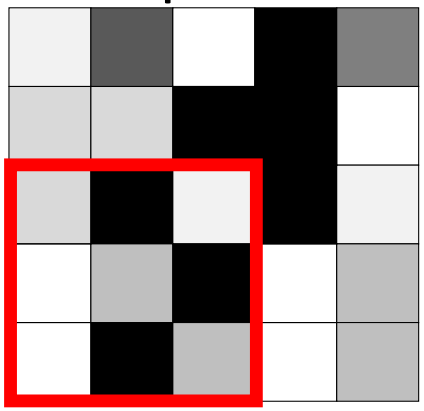
$f(x) = w^T x$

Feature map



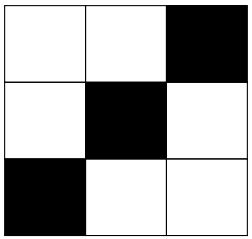
Let's make that stride = 2

Input X

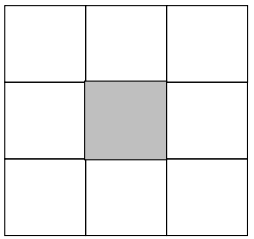


The shared weights W

(AKA a filter, AKA a feature)



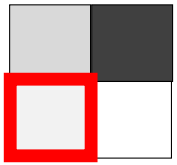
Filtered values



A shared activation function

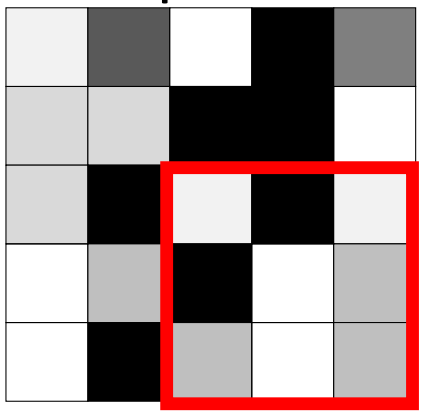
$f(x) = w^T x$

Feature map



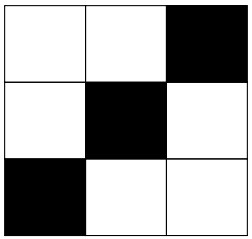
Let's make that stride = 2

Input X

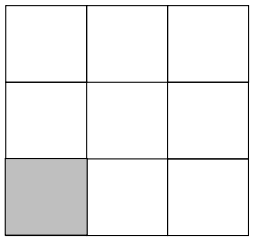


The shared weights W

(AKA a filter, AKA a feature)



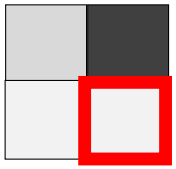
Filtered values



A shared activation function

$f(x) = w^T x$

Feature map

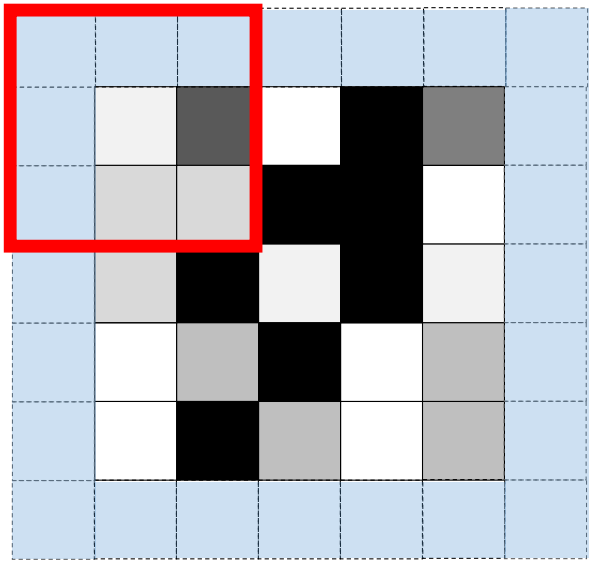


Padding

- Extra blank rows and columns added around your input.

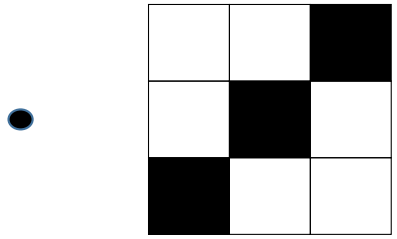
Stride = 2, Padding = 1

Input X

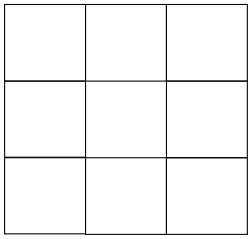


The shared weights W

(AKA a filter, AKA a feature)



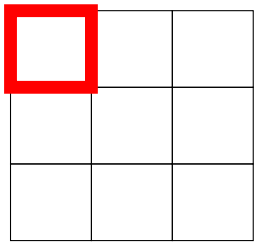
Filtered values



A shared activation function

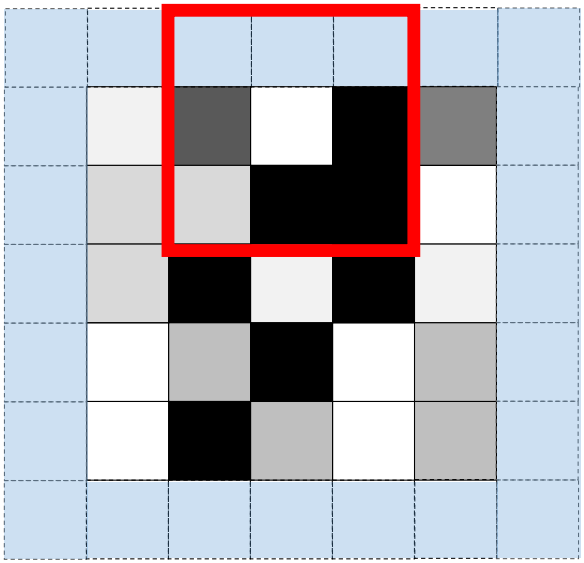
$f(x) = w^T x$

Feature map



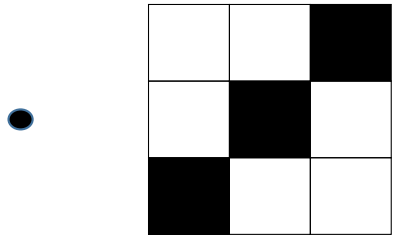
Stride = 2, Padding = 1

Input X

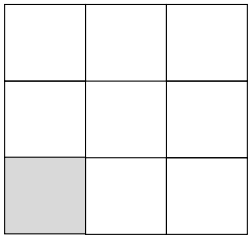


The shared weights W

(AKA a filter, AKA a feature)



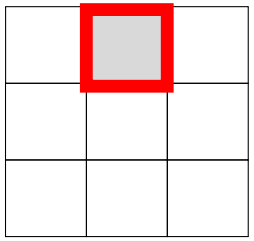
Filtered values



A shared activation function

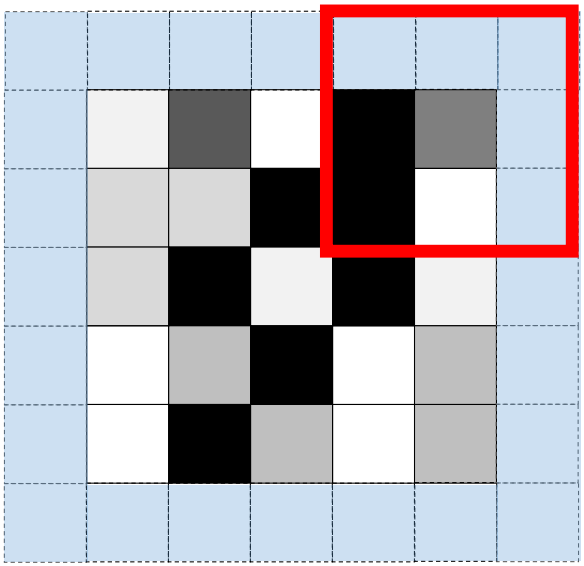
$f(x) = w^T x$

Feature map



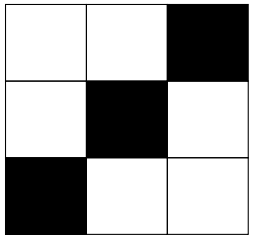
Stride = 2, Padding = 1

Input X

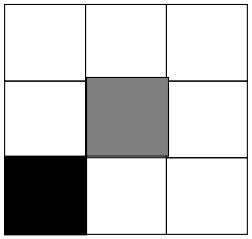


The shared weights W

(AKA a filter, AKA a feature)



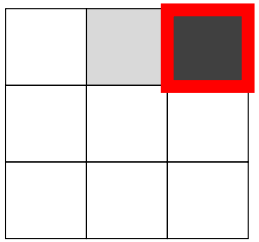
Filtered values



A shared activation function

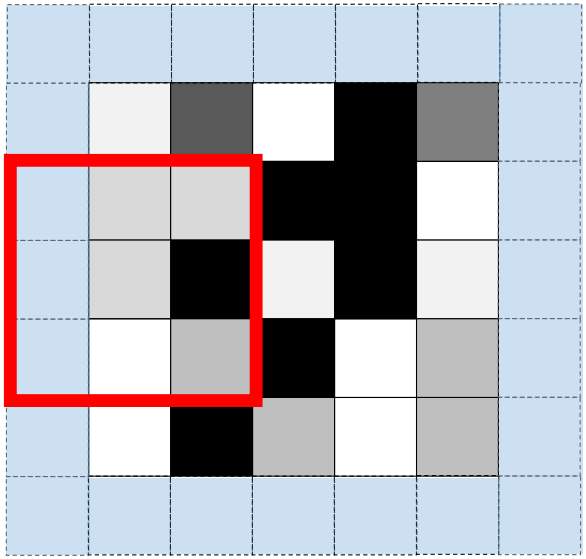
$f(x) = w^T x$

Feature map



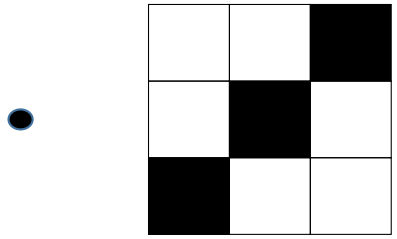
Stride = 2, Padding = 1

Input X

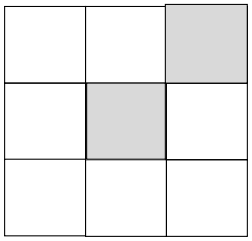


The shared weights W

(AKA a filter, AKA a feature)



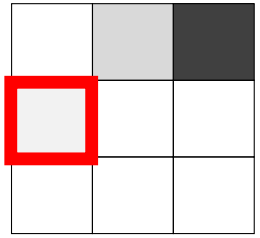
Filtered values



A shared activation function

$f(x) = w^T x$

Feature map

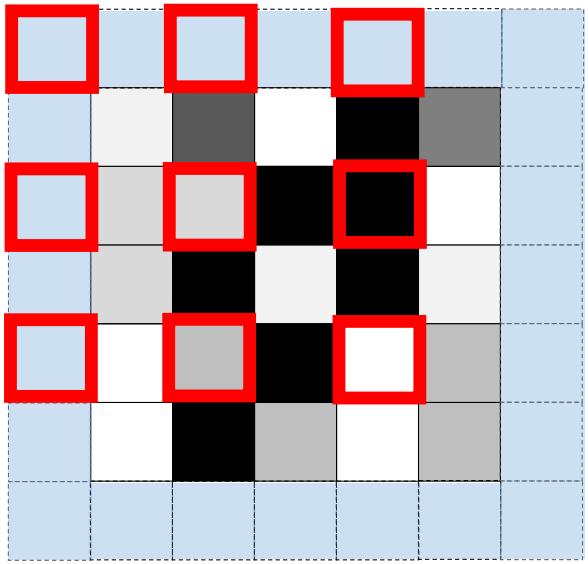


Dilation

- Space out the squares of the filter on your input.

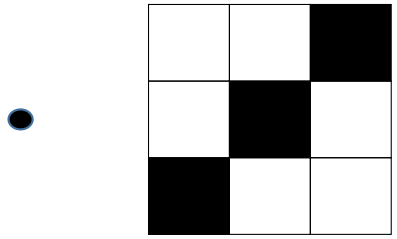
Stride = 2, Padding = 1, Dilation = 2

Input X

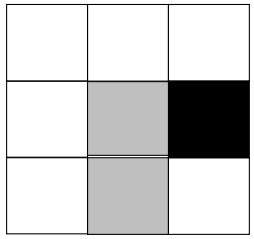


The shared weights W

(AKA a filter, AKA a feature)



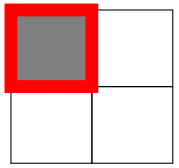
Filtered values



A shared activation function

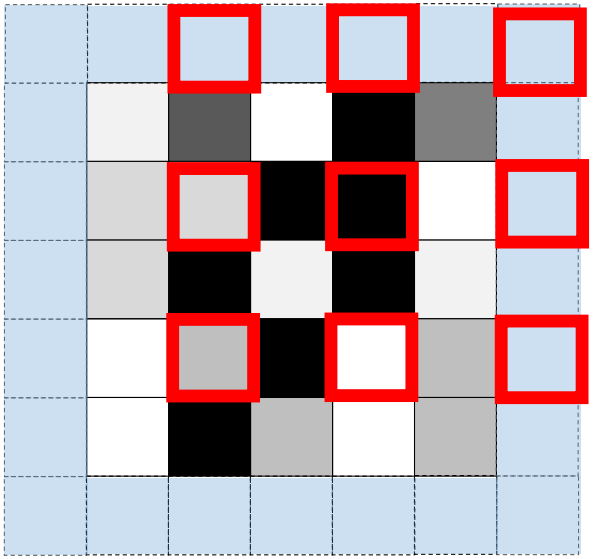
$f(x) = w^T x$

Feature map



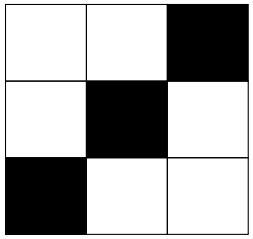
Stride = 2, Padding = 1, Dilation = 2

Input X

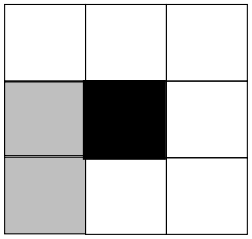


The shared weights W

(AKA a filter, AKA a feature)



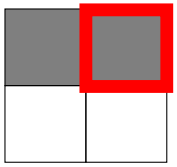
Filtered values



A shared activation function

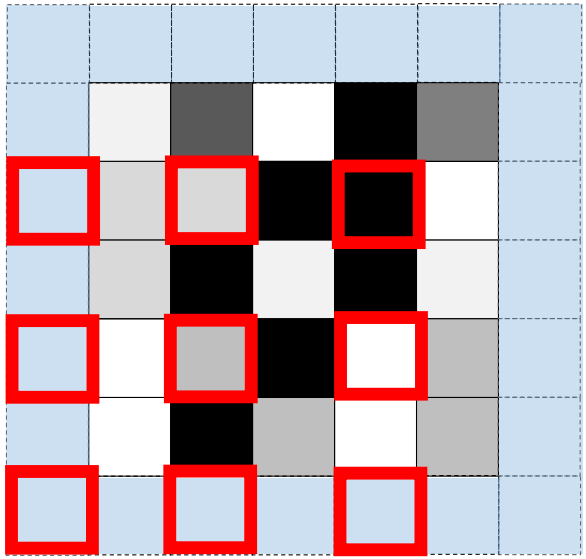
$f(x) = w^T x$

Feature map



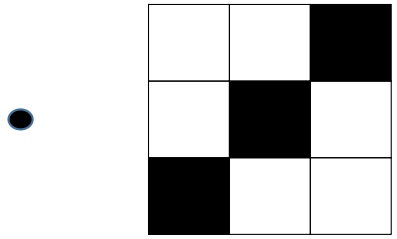
Stride = 2, Padding = 1, Dilation = 2

Input X

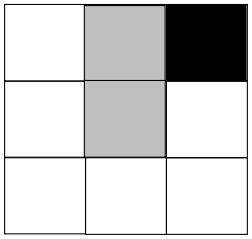


The shared weights W

(AKA a filter, AKA a feature)



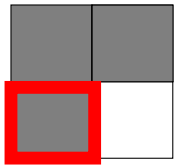
Filtered values



A shared activation function

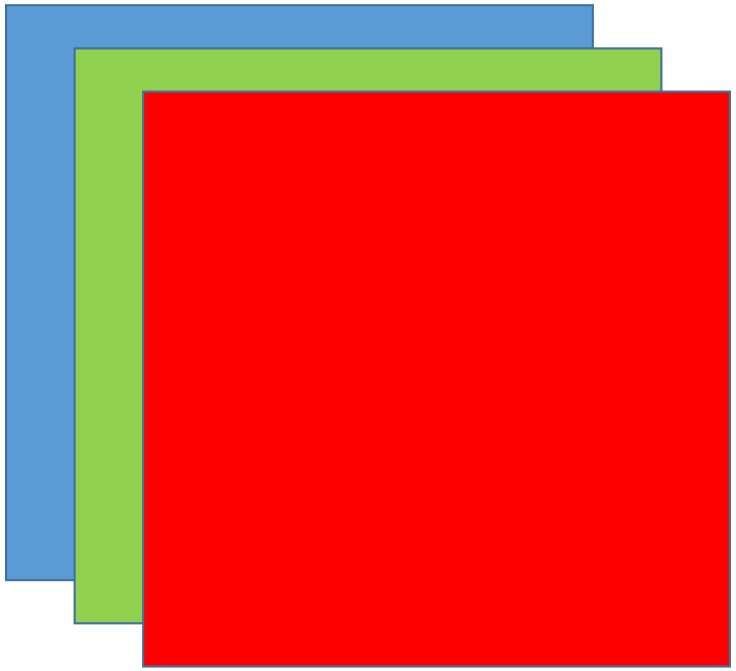
$f(x) = w^T x$

Feature map

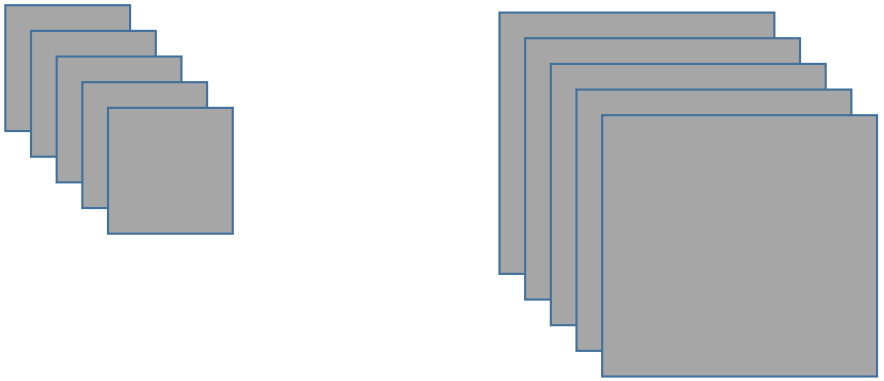


Channels

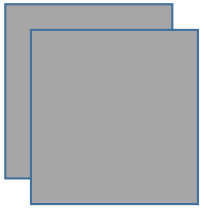
RGB 3-color input has 3 channels



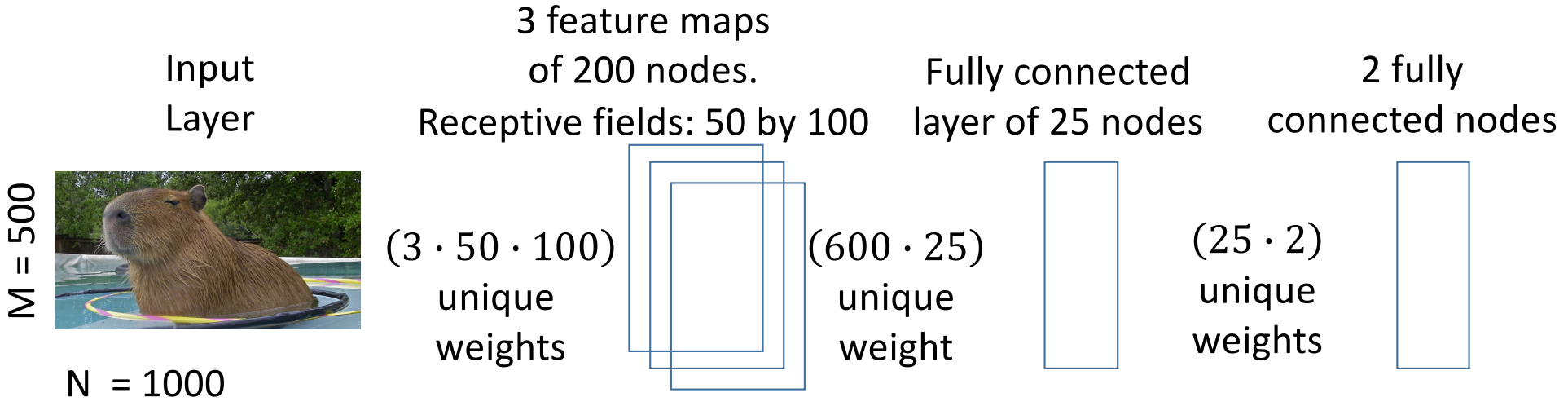
Convolutional layer with 5 channel output



Convolutional layer with 2 channel output



How many weights in a convolutional net?



$15{,}000 + 15{,}000 + 50 = 30{,}050$ unique weights

Compare that to the 50,005,100 weights in the other network

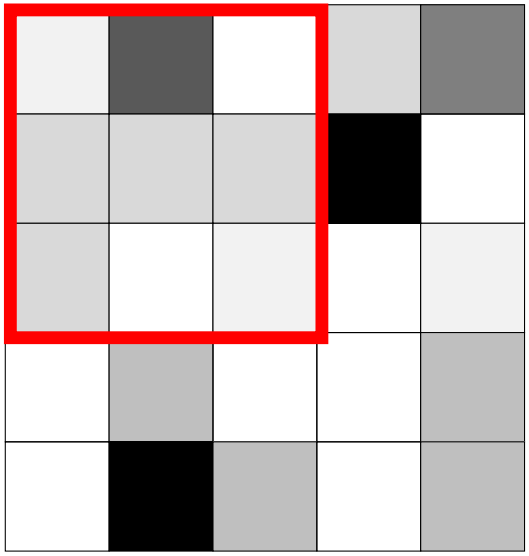
Is that enough reduction?

- That picture of the adorable Capybara was 500,000 pixels.
- The 2017 iPhone X takes 12 megapixel images. That's 24 times as big.
- Making the network on the previous slide 24 times bigger would have us at over 600,000 weights.
- Can we do some kind of down sampling on our data?

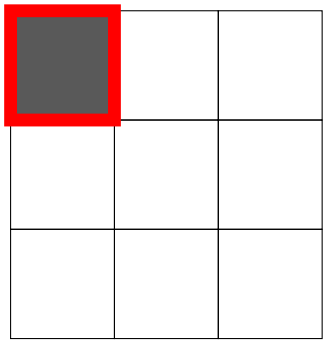
Max Pool Layer: A kind of downsampling

- Max Pool $f(x) = \max(x_1, x_2, \ldots x_n)$

LAYER N



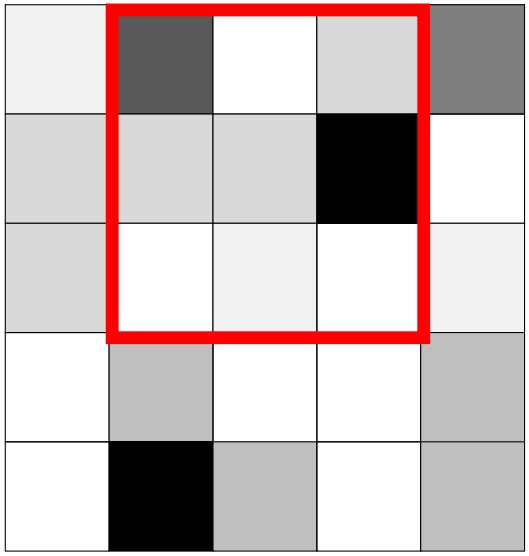
LAYER N+1



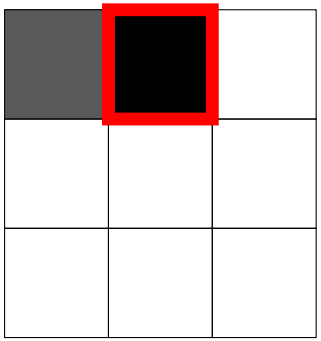
Max Pool Layer: A kind of downsampling

- Max Pool $f(x) = \max(x_1, x_2, \ldots x_n)$

LAYER N

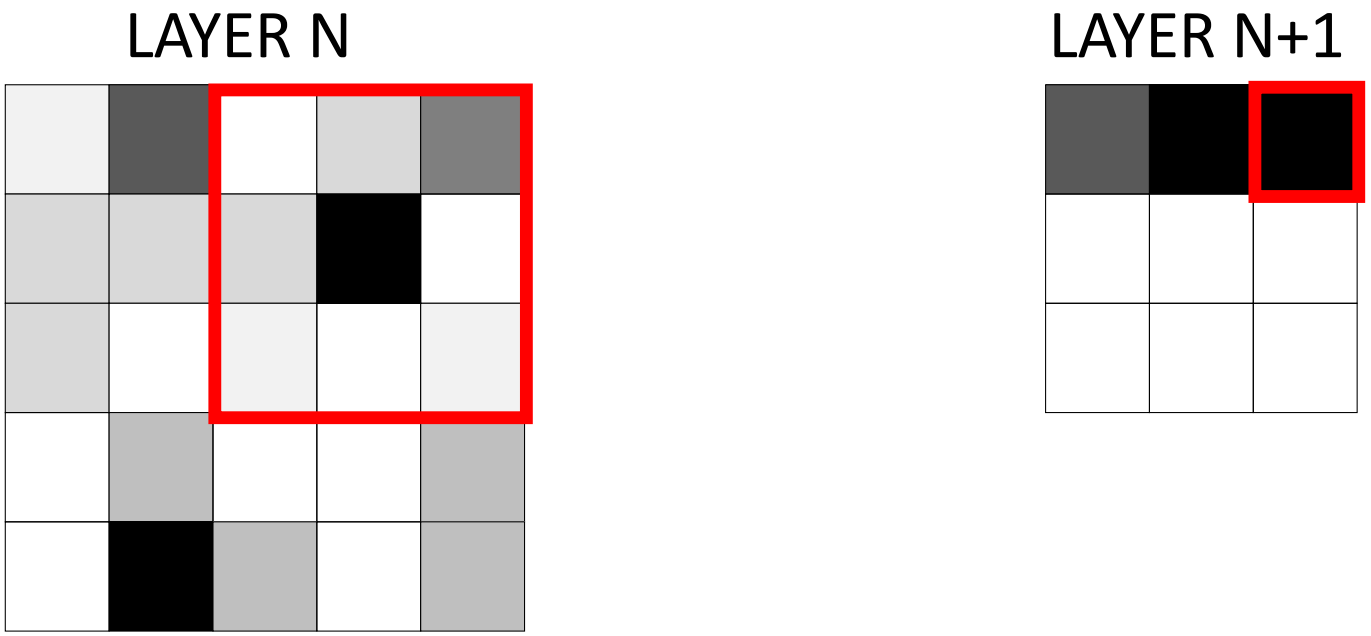


LAYER N+1



Max Pool Layer: A kind of downsampling

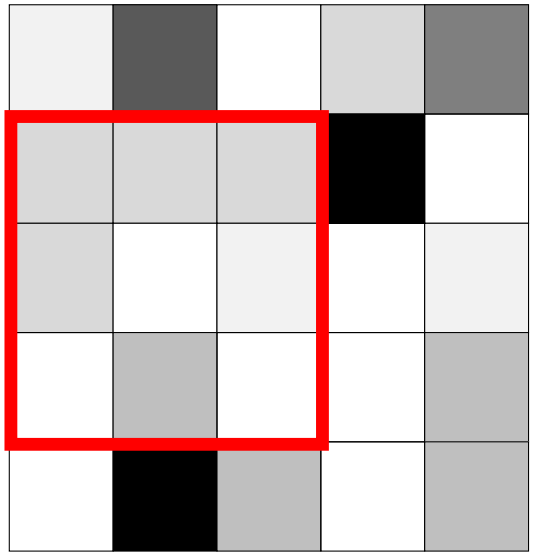
- Max Pool $f(x) = \max(x_1, x_2, \ldots x_n)$



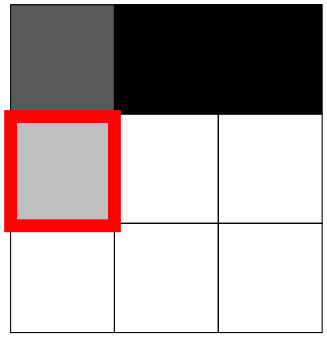
Max Pool Layer: A kind of downsampling

- Max Pool $f(x) = \max(x_1, x_2, \ldots x_n)$

LAYER N



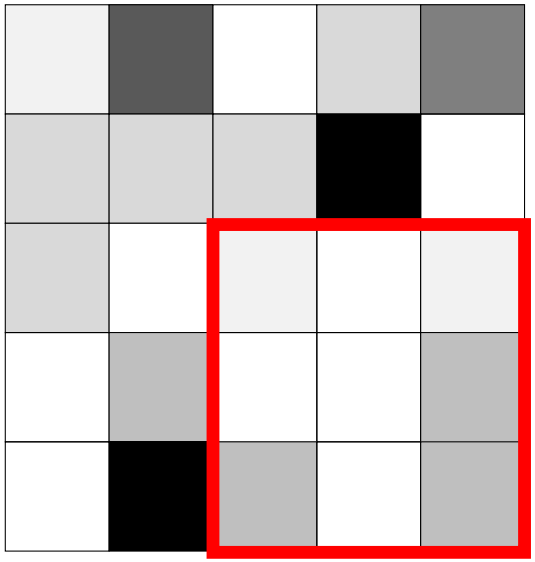
LAYER N+1



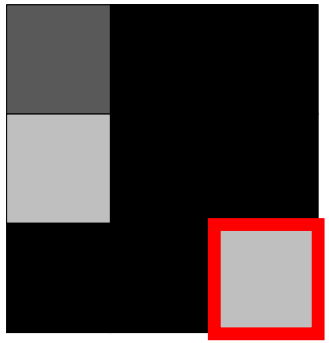
Max Pool Layer: A kind of downsampling

- Max Pool $f(x) = \max(x_1, x_2, \ldots x_n)$

LAYER N



LAYER N+1



Other kinds of pooling

- Min pooling
- Average pooling
- When would you want to use each of these? How would you pick?

Reduce your patch size, if you can

- Use a small patch of the spectrogram as input (e.g. 100 by 100 patch of the spectrogram)
- Reduces the number of model parameters needed
- Increases the number of training examples

1000 Spectrograms * 600 patches* 100augmentations = 60 million

So…what is a convolutional net?

- A network with one or more layers that are feature maps
- A layer with feature maps is called a “convolutional layer”
- Often, convolutional layers are alternated with pooling layers.
- Since these nets have many fewer connections
  - They train faster
  - They need fewer training examples