Lecture 19
Convolutional Neural Networks

Disclosure: The slides are adopted from Stanford CS231n and Andrew Ng's online Machine Learning Course.

Research Questions: Visual Recognition

Research Questions: Visual Recognition

Emotion Detection

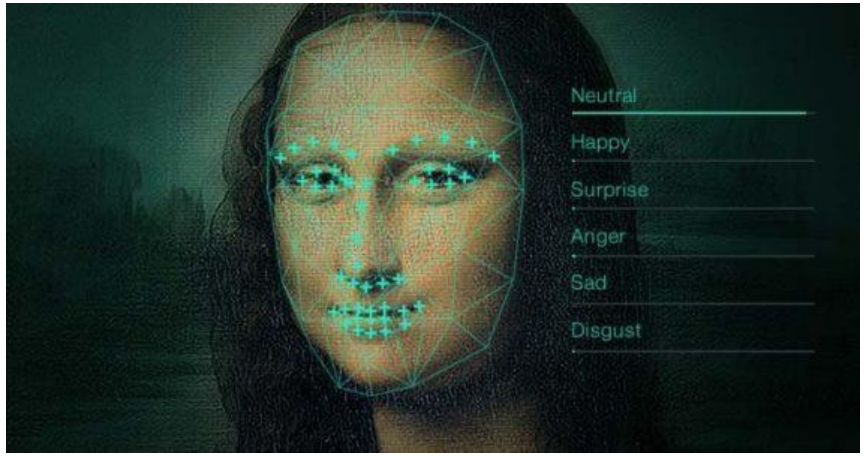


Image Source: https://www.cbronline.com/news/automatic-facial-recognition

Research Questions: Visual Recognition

Emotion Detection

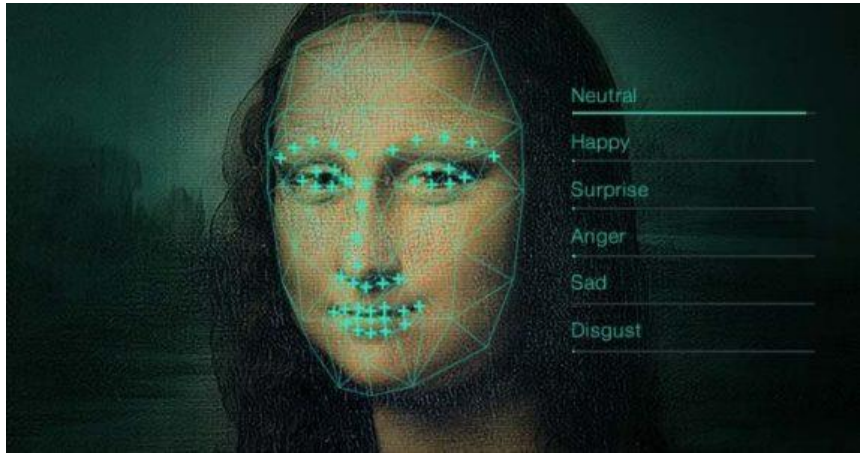


Image Source: https://www.cbronline.com/news/automatic-facial-recognition

Object Detection

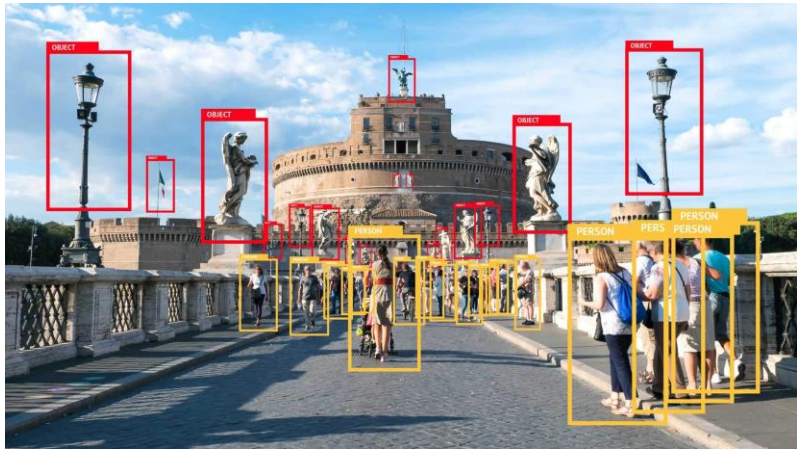


Image Source: https://martechtoday.com/how-visual-recognition-is-set-to-change-advertising-223719

Research Questions: Visual Recognition

Emotion Detection

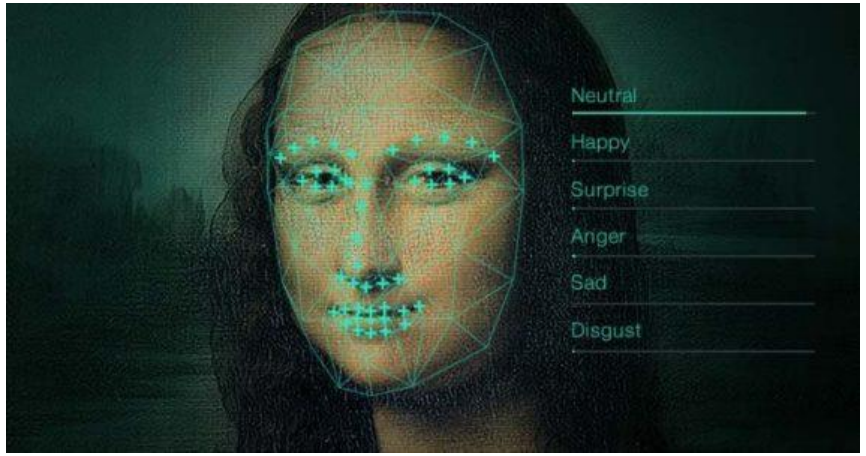


Image Source: https://www.cbronline.com/news/automatic-facial-recognition

Object Detection

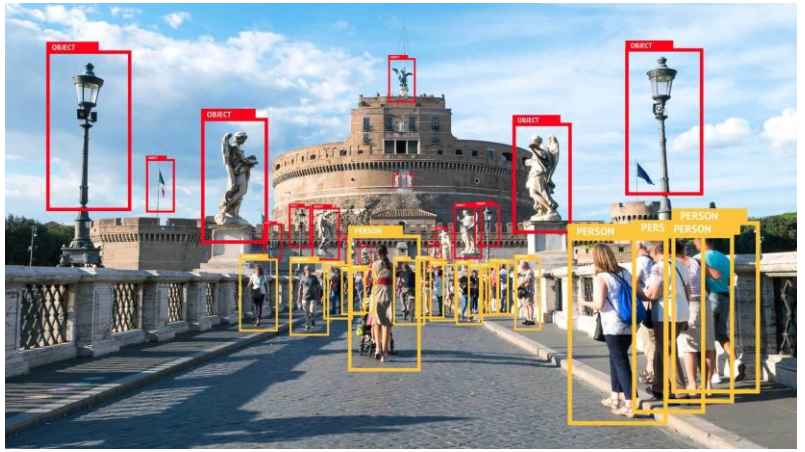


Image Source: https://martechtoday.com/how-visual-recognition-is-set-to-change-advertising-223719

Image Segmentation

Image Source: https://www.vox.com/future-perfect/2019/4/19/18412674/ai-bias-facial-recognition-black-gay-transgender

Research Questions: Visual Recognition

Emotion Detection

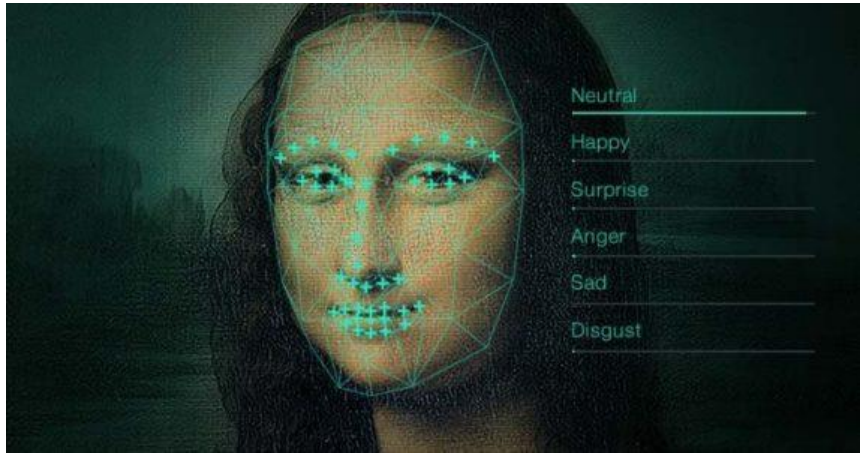


Image Source: https://www.cbronline.com/news/automatic-facial-recognition

Object Detection

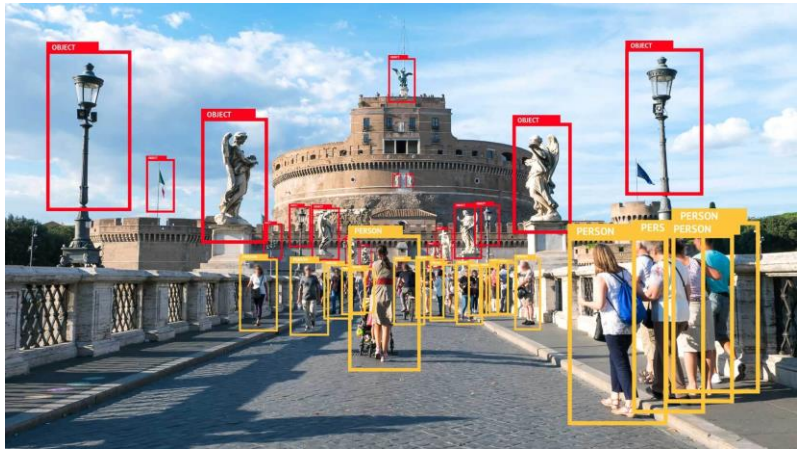


Image Source: https://martechtoday.com/how-visual-recognition-is-set-to-change-advertising-223719

Image Segmentation

Image Source: https://www.vox.com/future-perfect/2019/4/19/18412674/ai-bias-facial-recognition-black-gay-transgender

Image Captioning

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

Image Source: https://towardsdatascience.com/image-captioning-in-deep-learning-9cd23fb4d8d2

Research Questions: Visual Recognition

Emotion Detection

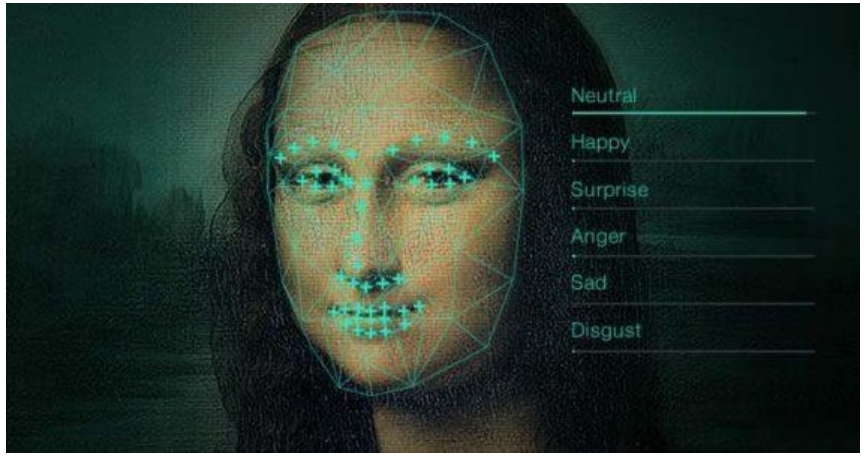


Image Source: https://www.cbronline.com/news/automatic-facial-recognition

Object Detection

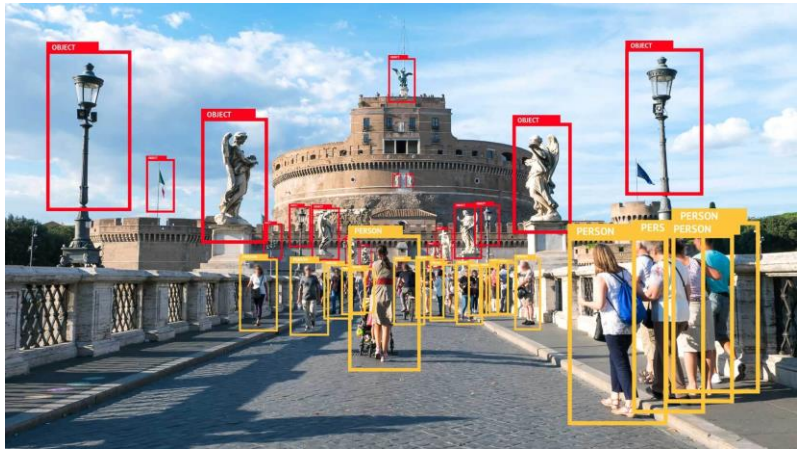


Image Source: https://martechtoday.com/how-visual-recognition-is-set-to-change-advertising-223719

Image Segmentation

Image Source: https://www.vox.com/future-perfect/2019/4/19/18412674/ai-bias-facial-recognition-black-gay-transgender

Image Captioning

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

Image Source: https://towardsdatascience.com/image-captioning-in-deep-learning-9cd23fb4d8d2

Pose Detection

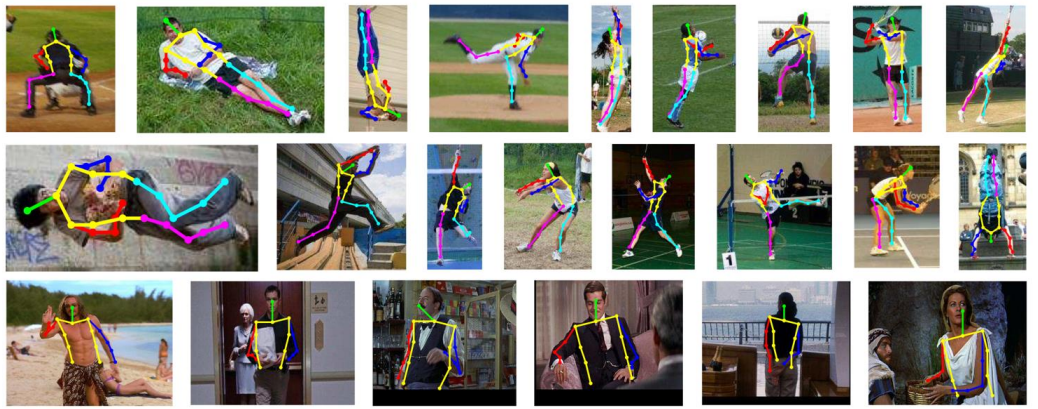
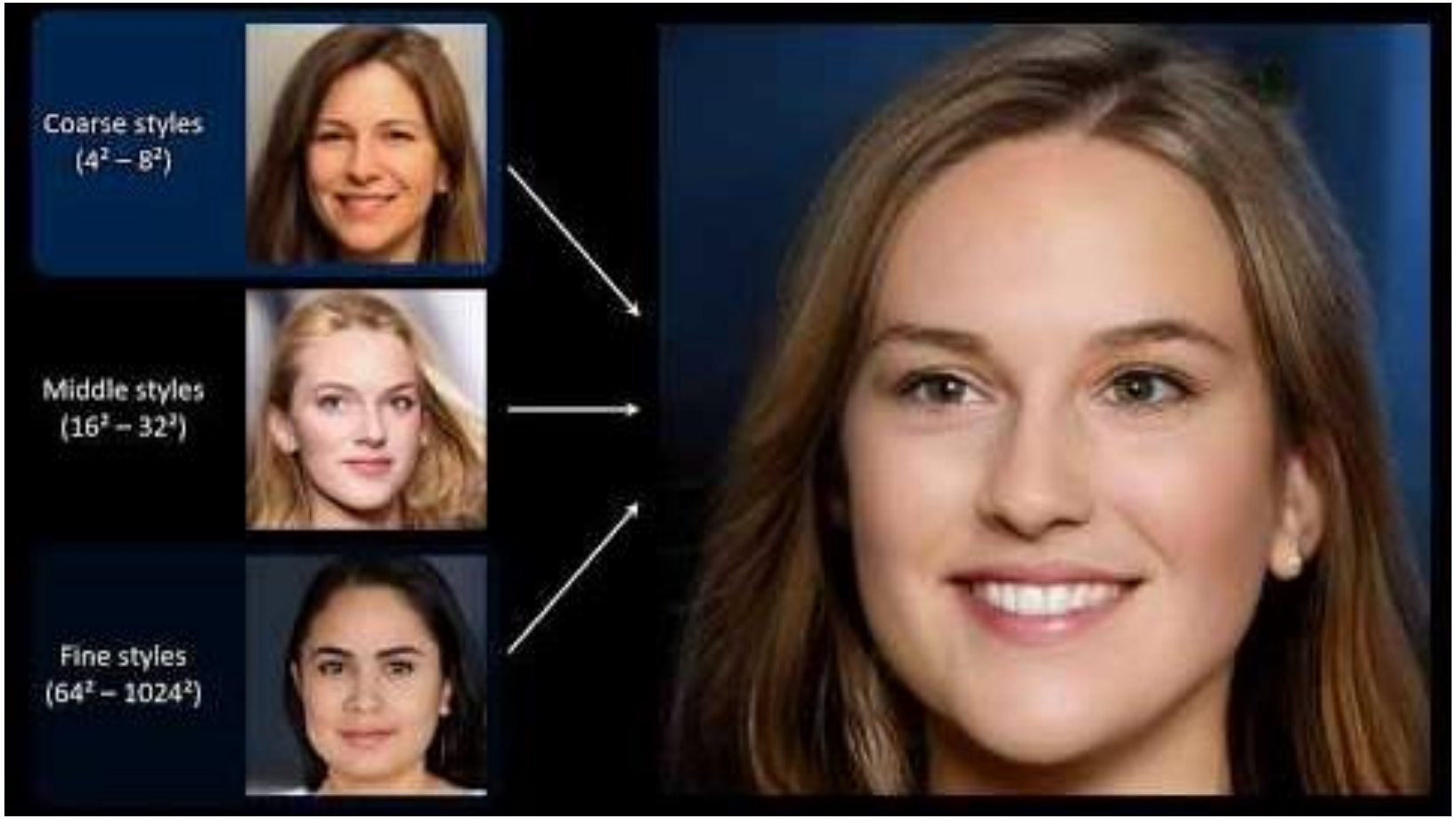


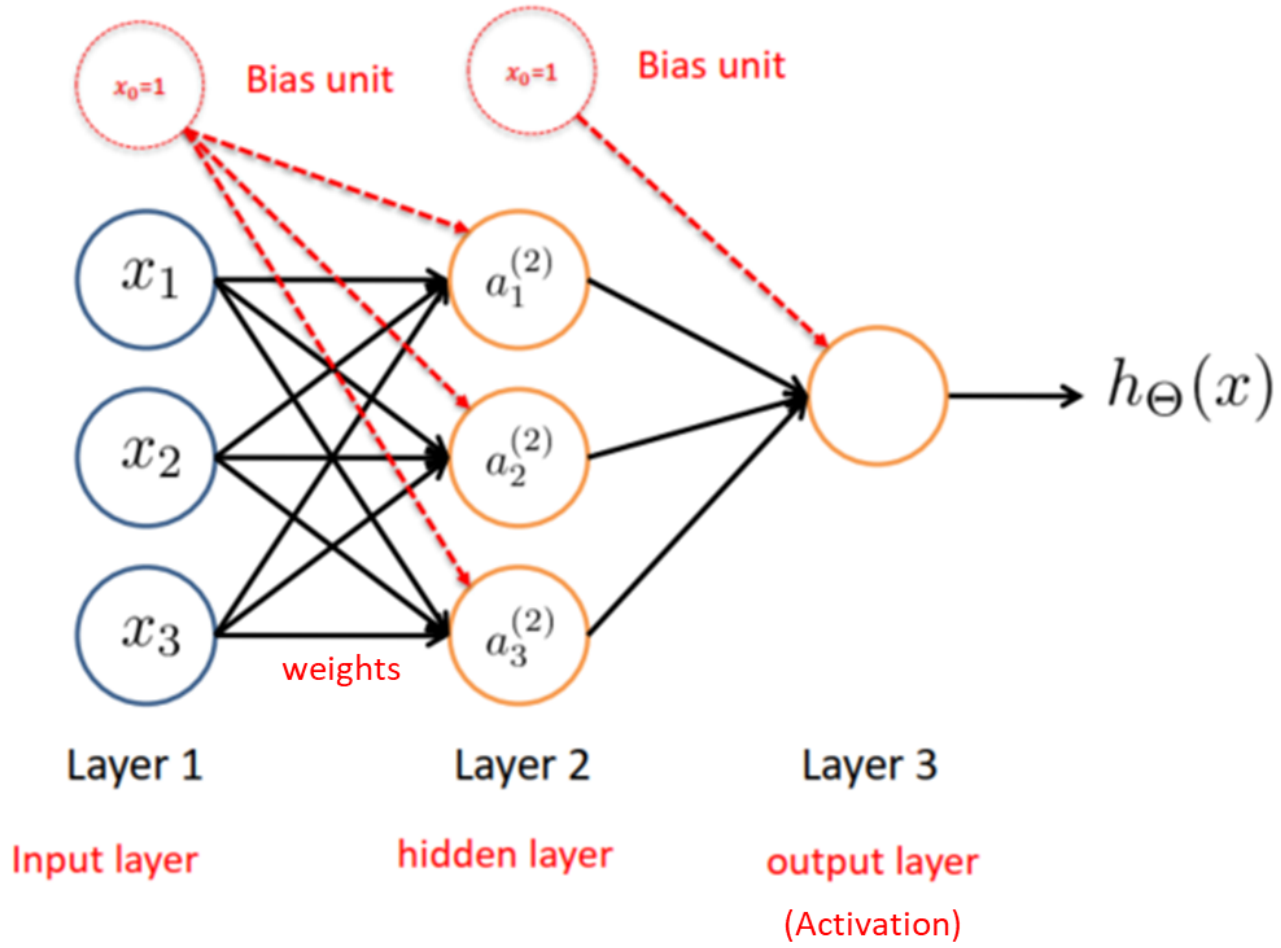
Image Source: https://nanonets.com/blog/human-pose-estimation-2d-guide/

Research Questions: Image Generation

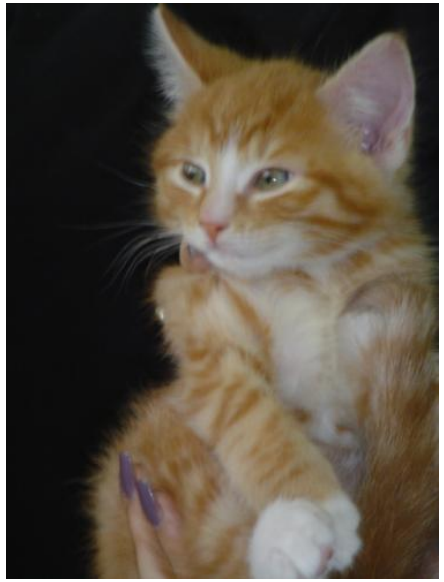


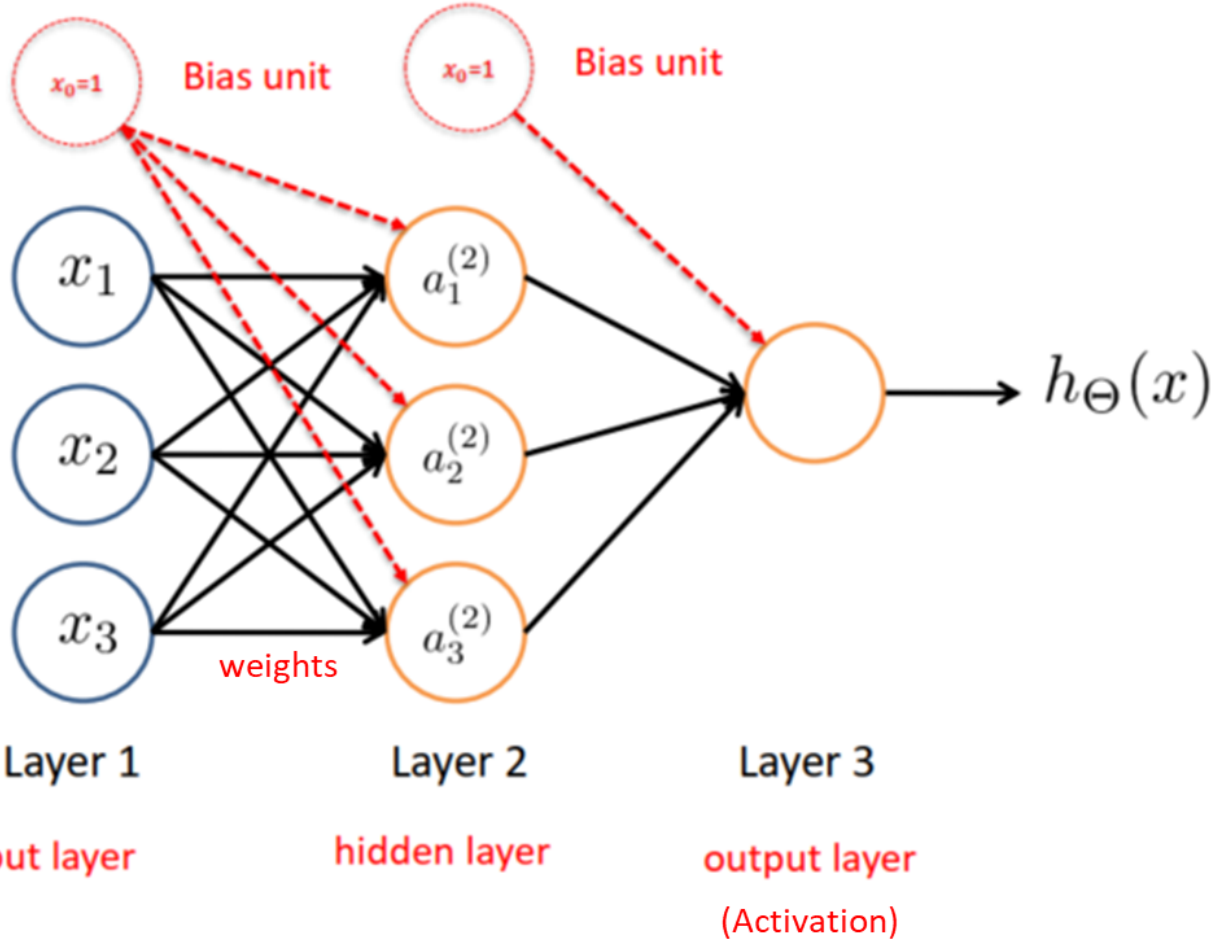
Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4401-4410. 2019.

Recall from last time

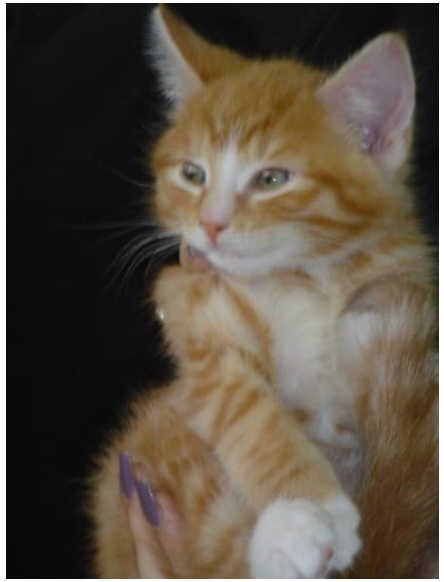


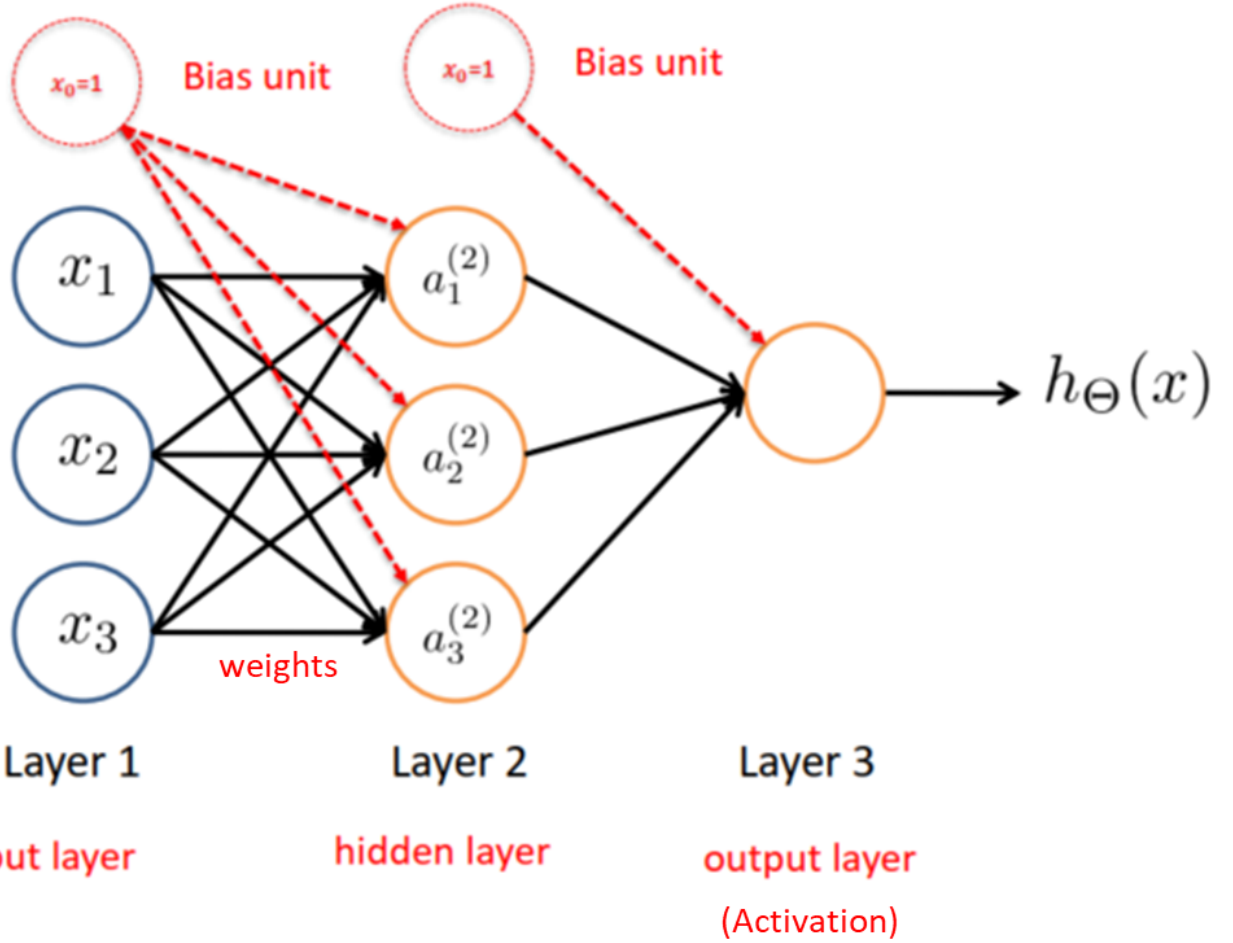
Recall from last time





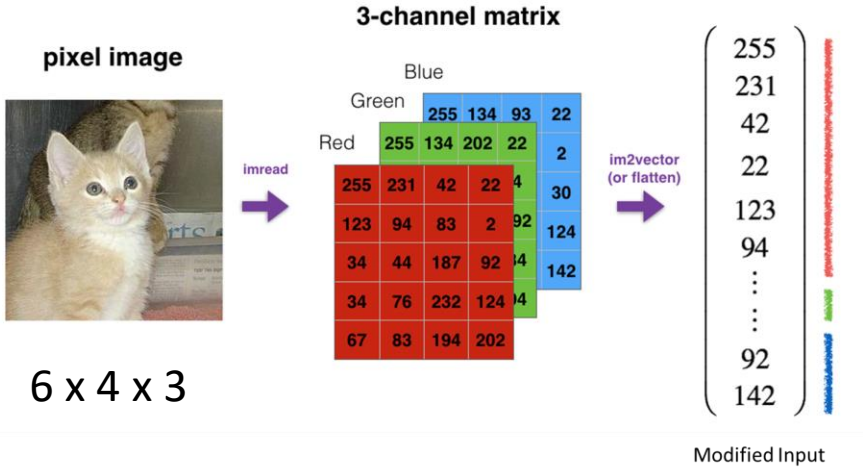
Recall from last time

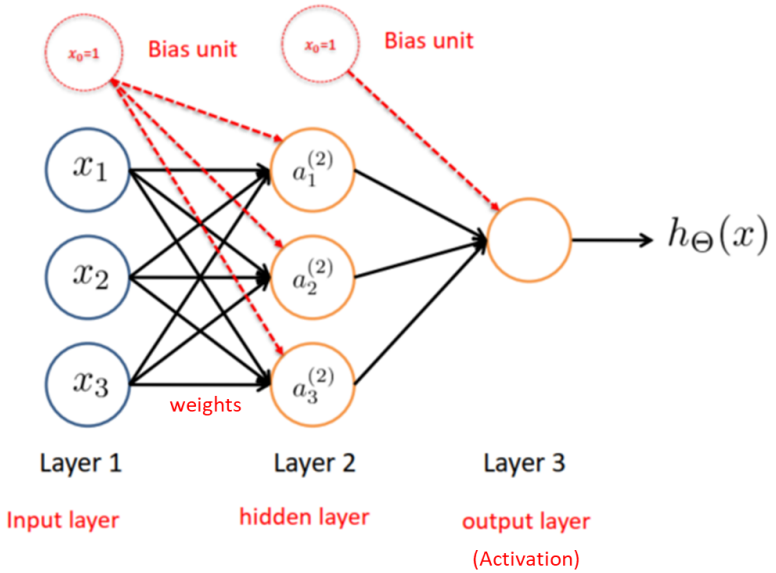




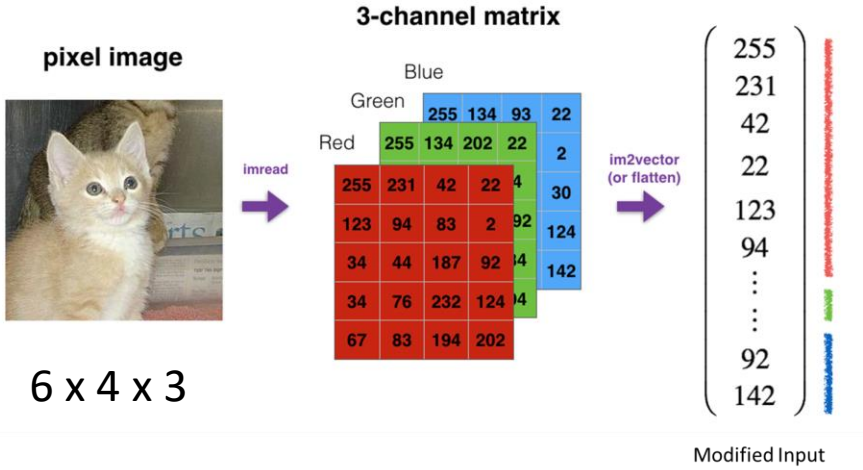
Cat

Fully Connected Layers

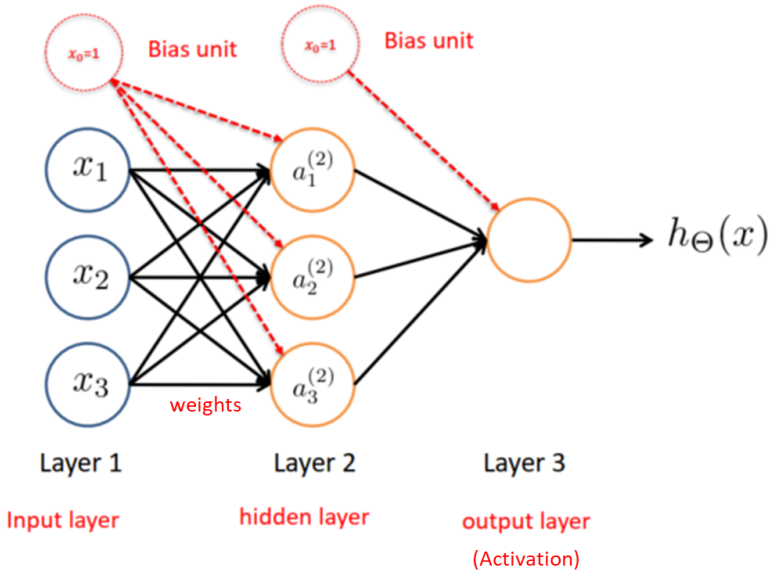


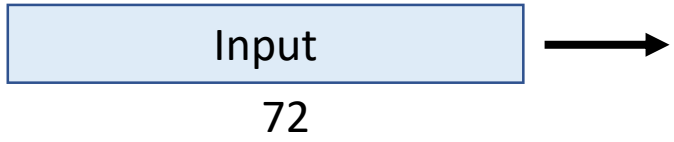


Fully Connected Layers

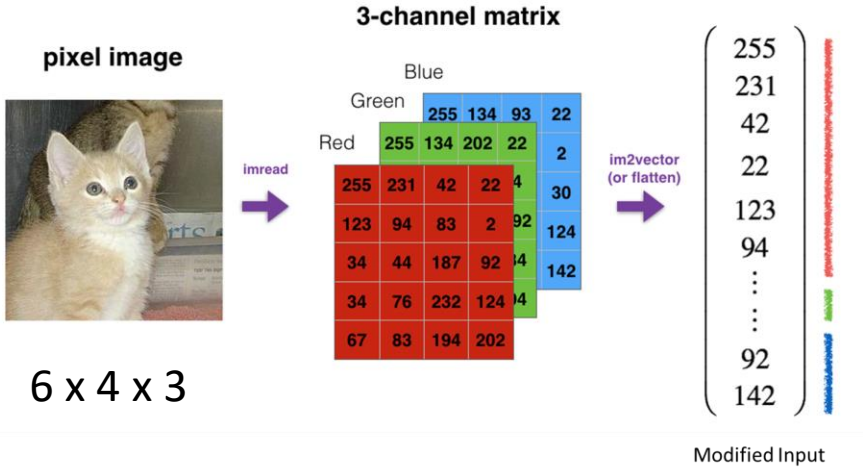


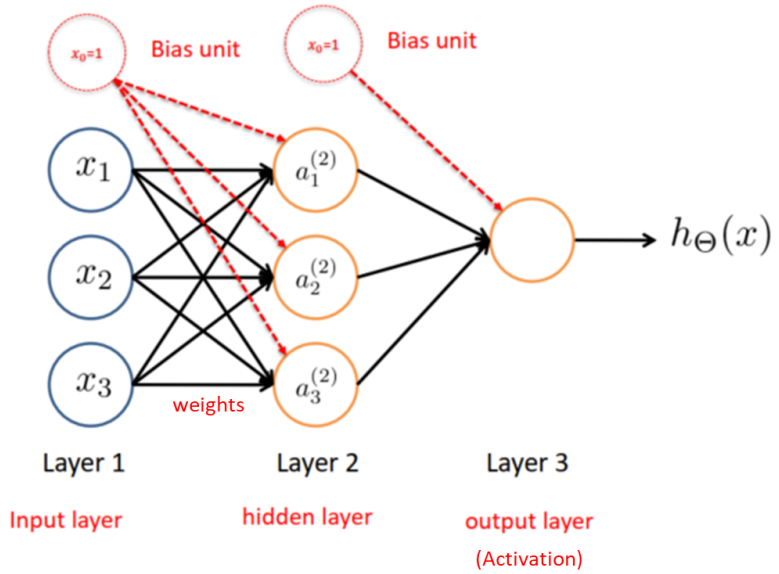
6 x 4 x 3





Fully Connected Layers





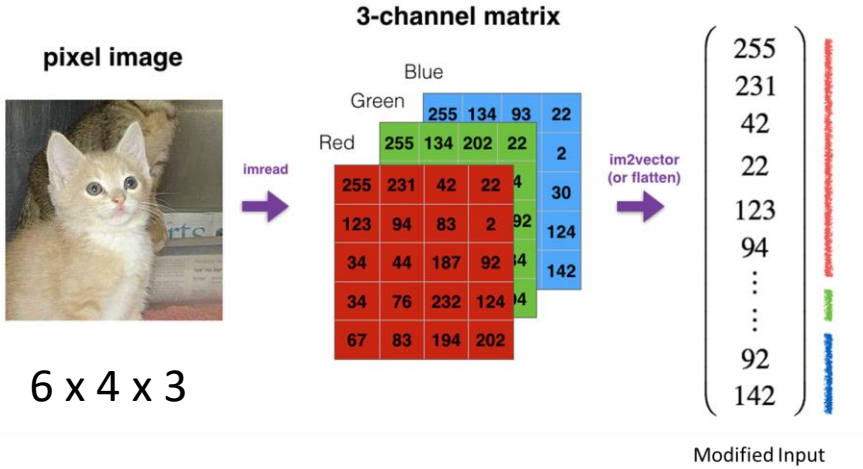
Input

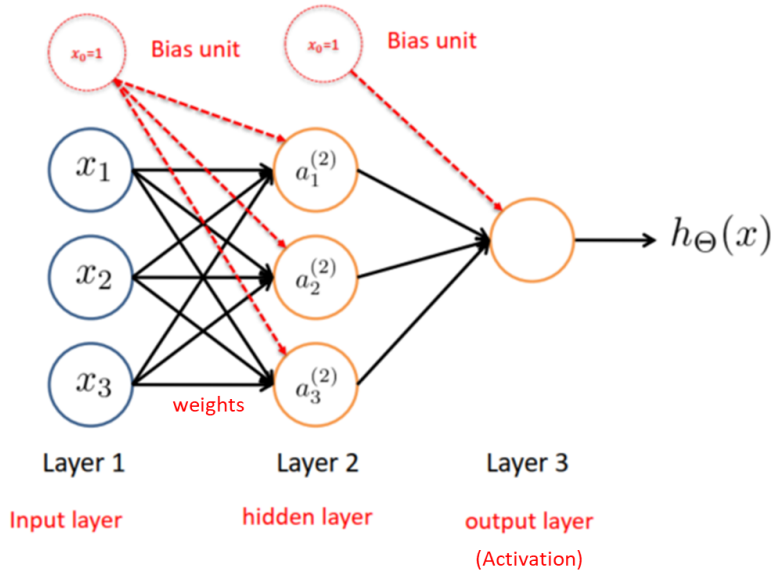
72

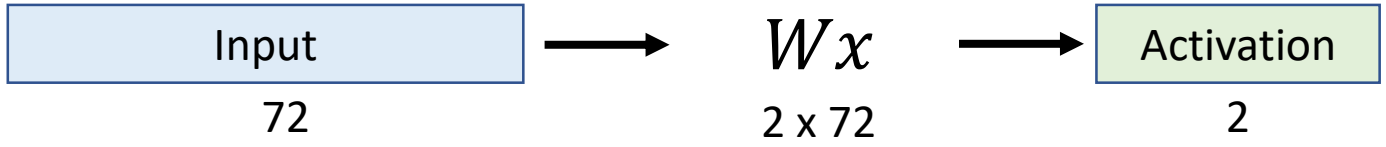
$$Wx$$

2 x 72

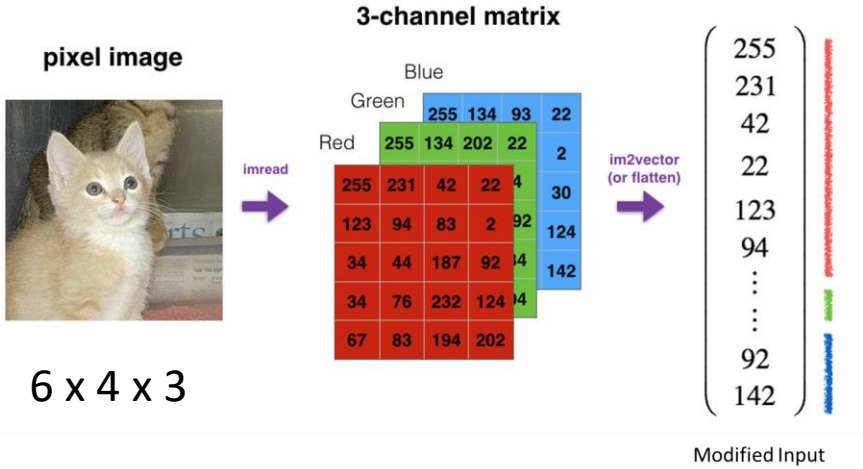
Fully Connected Layers

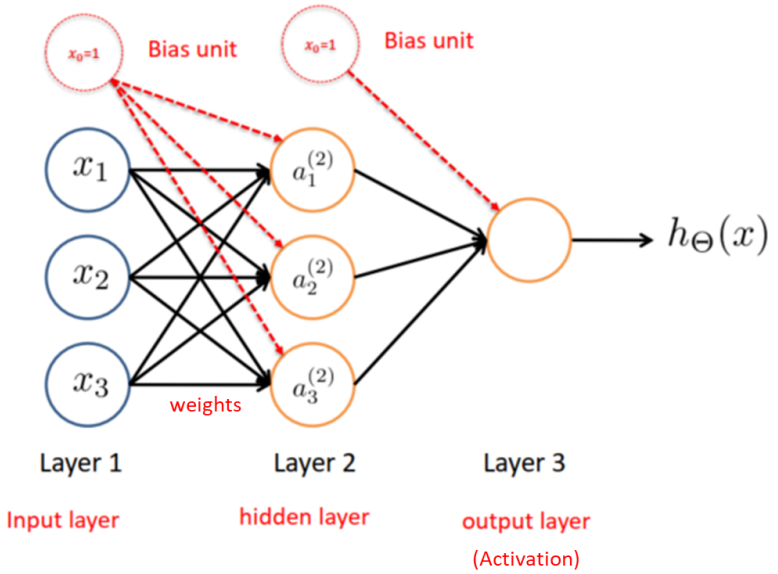






Fully Connected Layers





Input
72

Wx
2 x 72

Activation
2

Output: 1 Number
The result of taking a dot product between a row of W and the input
(a 72-dimensional dot product)

Issues in Image Recognition using Fully Connected Networks

Issues in Image Recognition using Fully Connected Networks

1. *They don’t scale well to full images*

Issues in Image Recognition using Fully Connected Networks

1. *They don't scale well to full images*

32 x 32 x 3 = 3072 weights

Width x Height x Color Channel

Issues in Image Recognition using Fully Connected Networks

1. *They don't scale well to full images*

32 x 32 x 3 = 3072 weights

Width x Height x Color Channel

What if we have a color image of 500 x 500 x 3
(750000 Weights)

For multiple layers, number of weights to calculate will Increase!

Issues in Image Recognition using Fully Connected Networks

1. They don't scale well to full images

32 x 32 x 3 = 3072 weights

Width x Height x Color Channel

What if we have a color image of 500 x 500 x 3
(750000 Weights)

For multiple layers, number of weights to calculate will Increase!

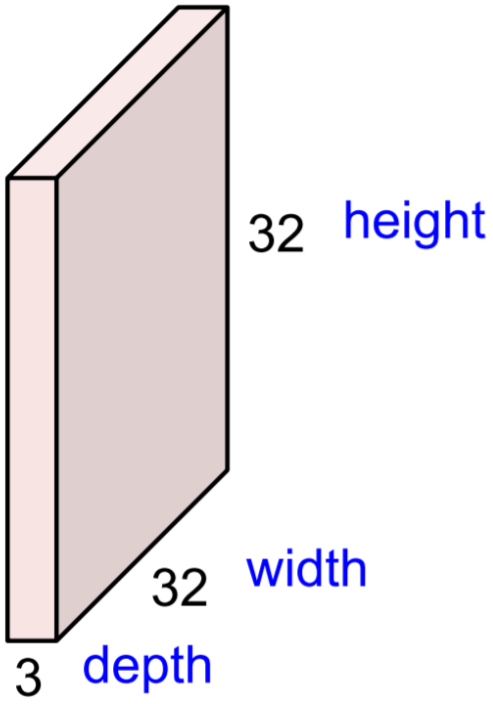
2. Doesn't preserve the spatial structure

Convolution Neural Networks

- We use three main types of layers to build Convolution Neural Network architectures
  1. **Convolutional Layer (**combined with activation function),
  2. **Pooling Layer**, and
  3. **Fully-Connected Layer (Exactly NN)**

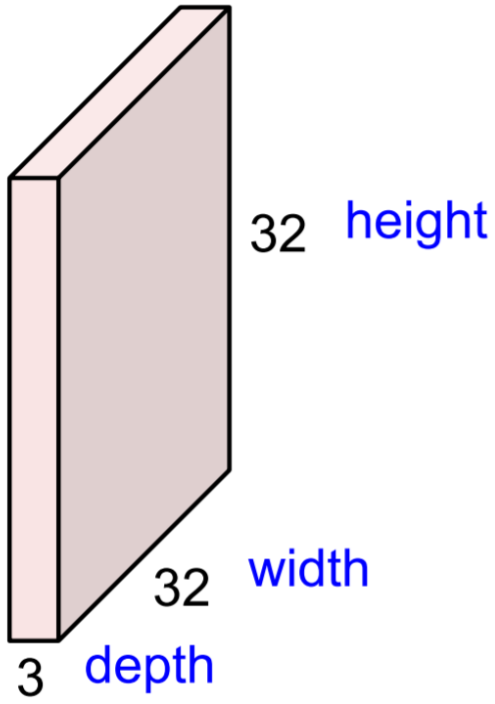
Convolution Layer (Higher Level Overview)

Convolution Layer (Higher Level Overview)

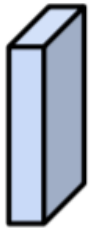


Input Image
32 x 32 x 3

Convolution Layer (Higher Level Overview)

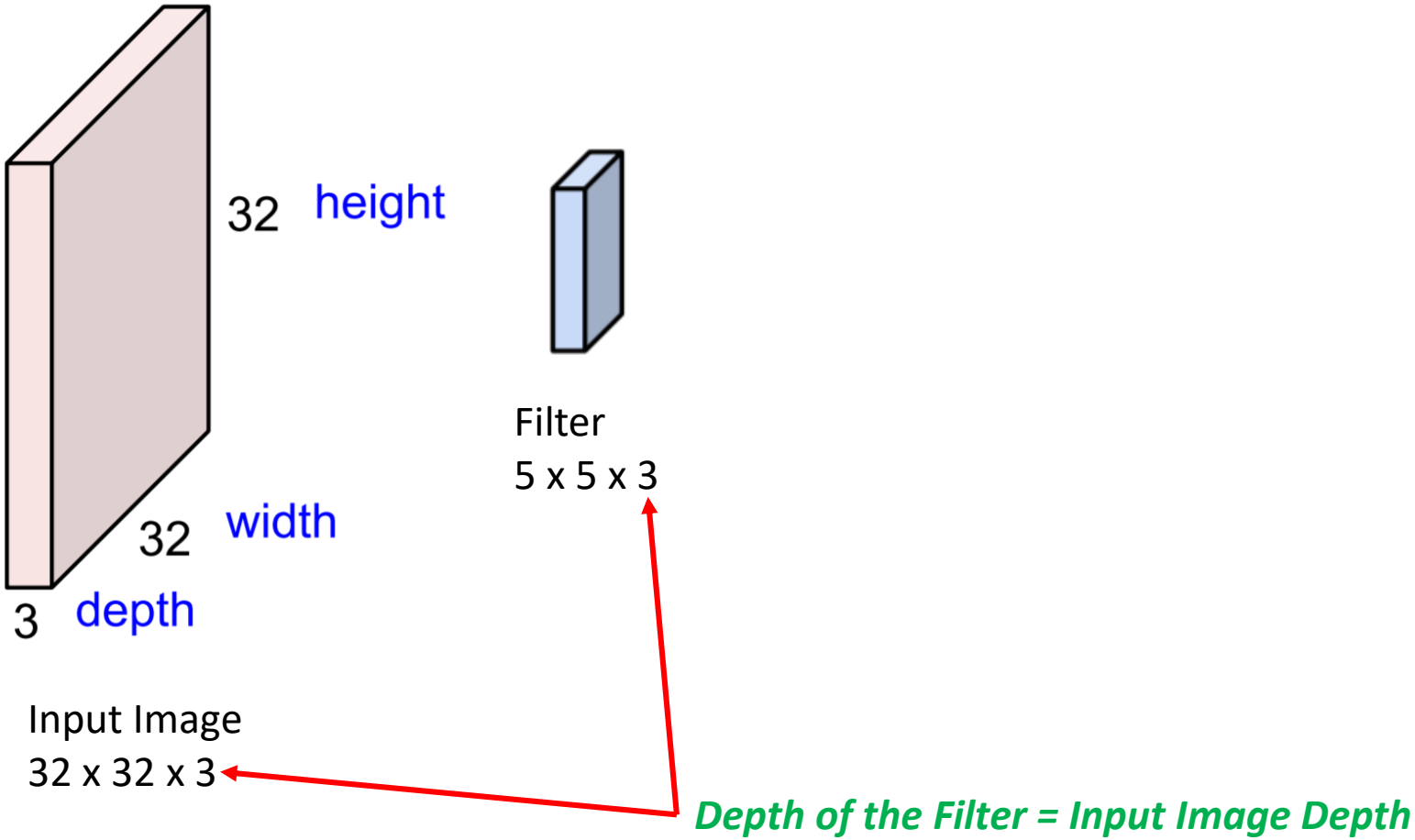


Input Image
32 x 32 x 3

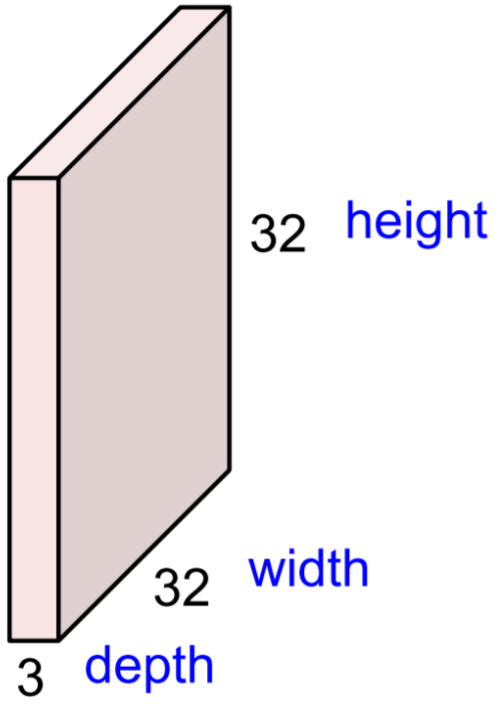


Filter
5 x 5 x 3

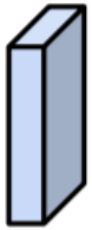
Convolution Layer (Higher Level Overview)



Convolution Layer (Higher Level Overview)

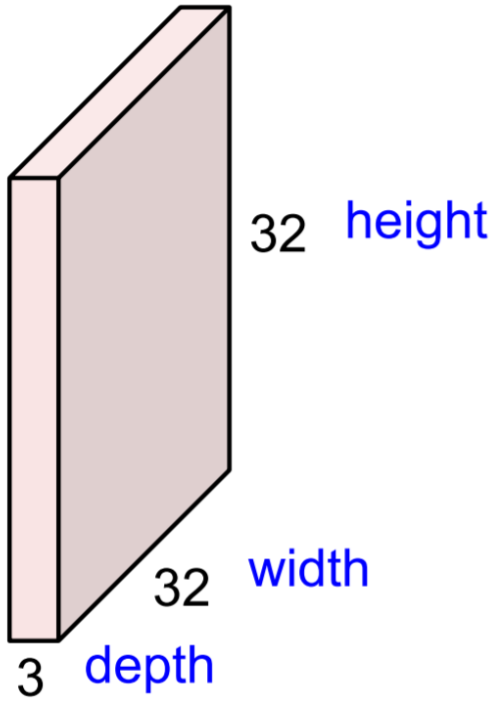


Input Image
32 x 32 x 3

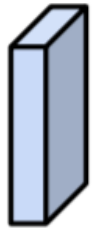


Filter
5 x 5 x 3

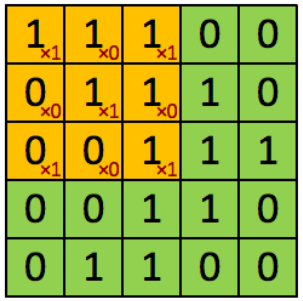
Convolution Layer (Higher Level Overview)



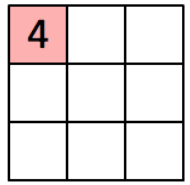
Input Image
32 x 32 x 3



Filter
5 x 5 x 3

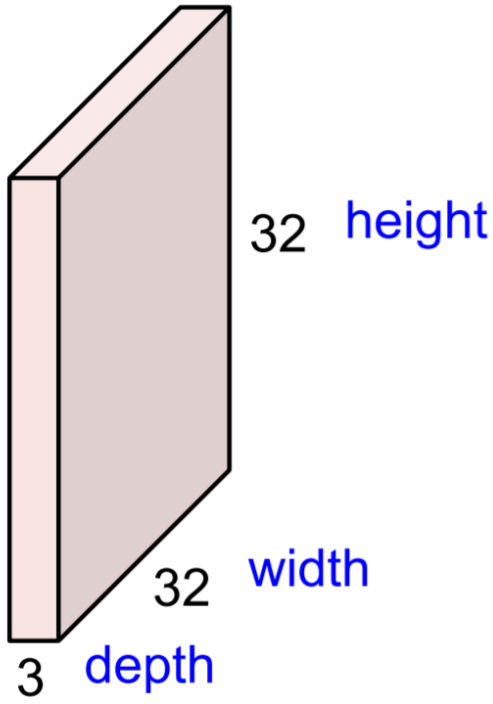


Image

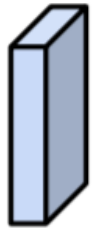


Convolved
Feature

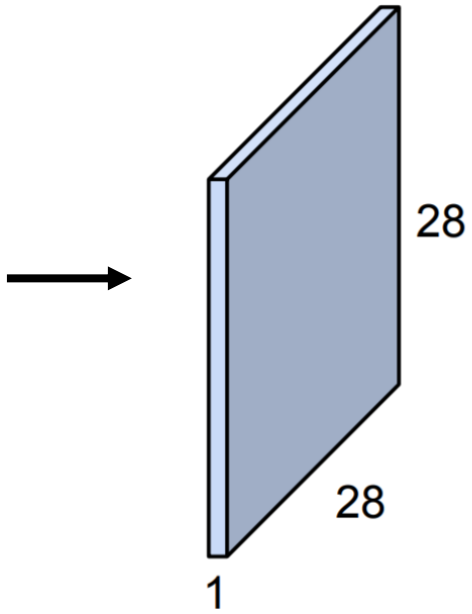
Convolution Layer (Higher Level Overview)



Input Image
32 x 32 x 3

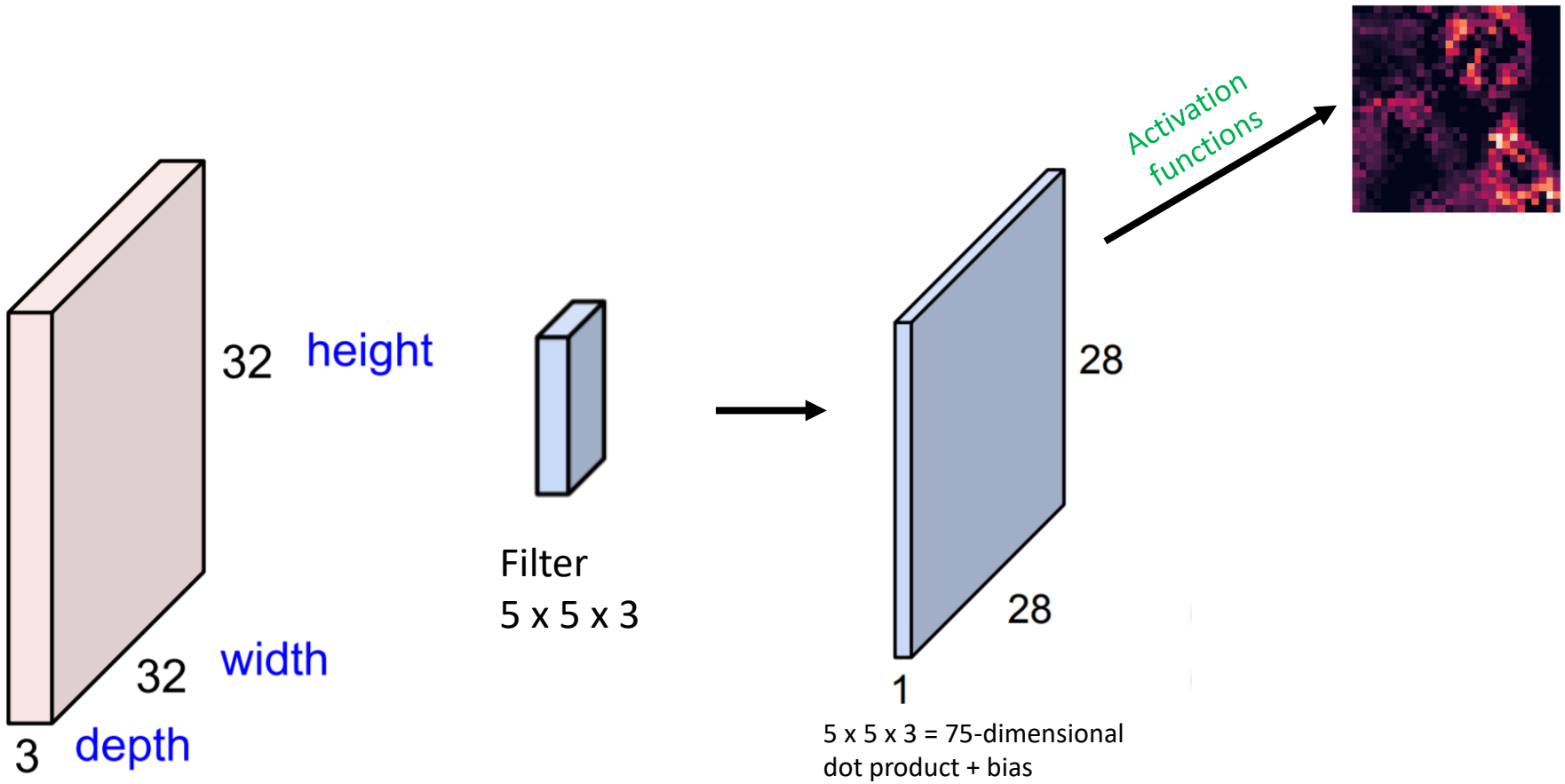


Filter
5 x 5 x 3

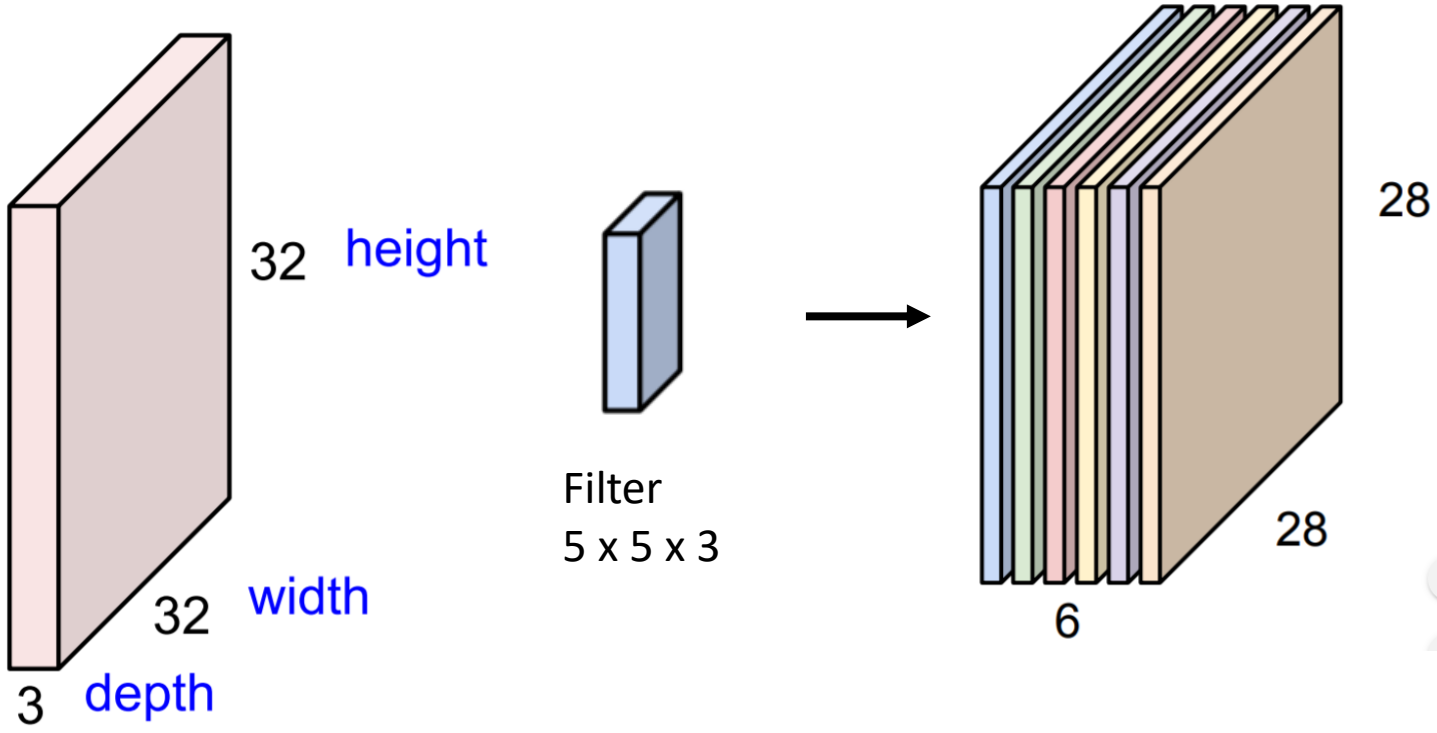


5 x 5 x 3 = 75-dimensional
dot product + bias

Convolution Layer (Higher Level Overview)



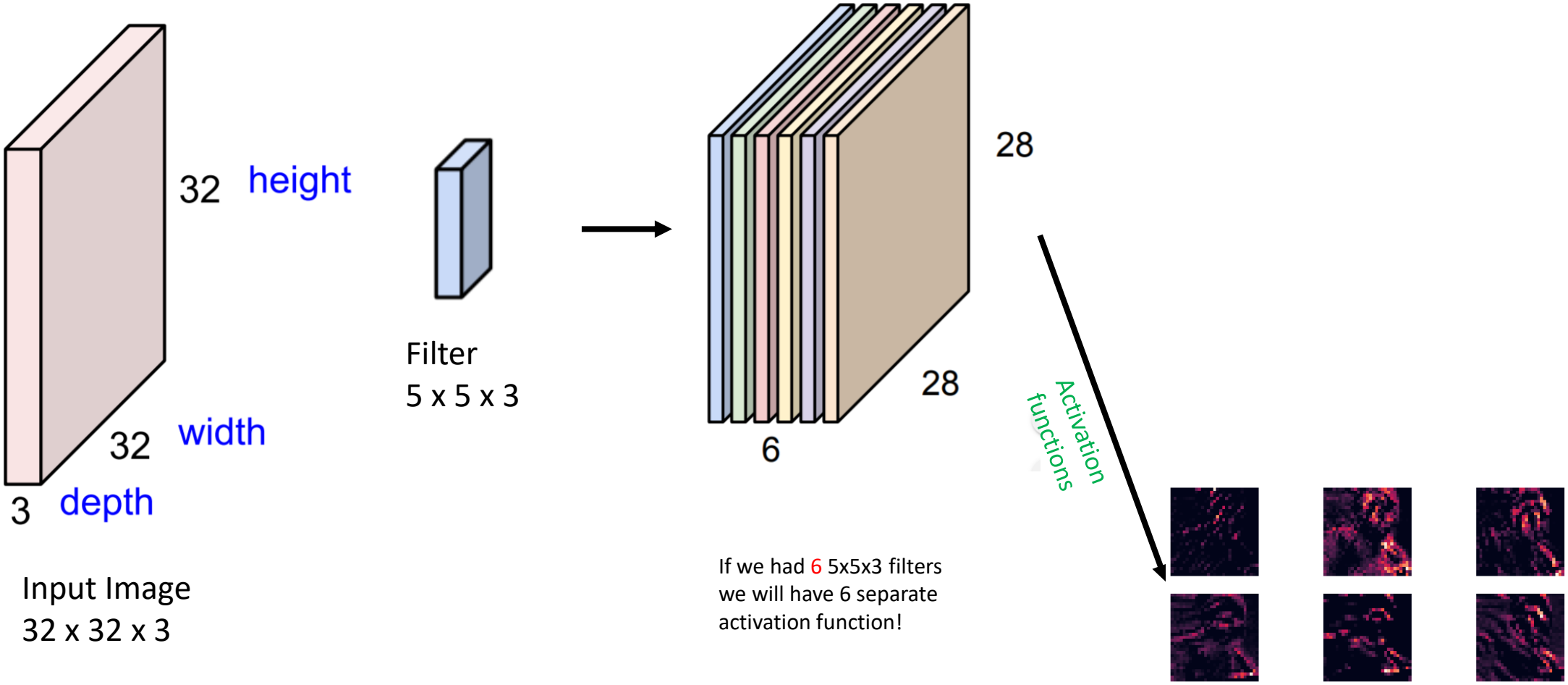
Convolution Layer (Higher Level Overview)



Input Image
32 x 32 x 3

If we had 6 5x5x3 filters
we will have 6 separate
activation function!

Convolution Layer (Higher Level Overview)



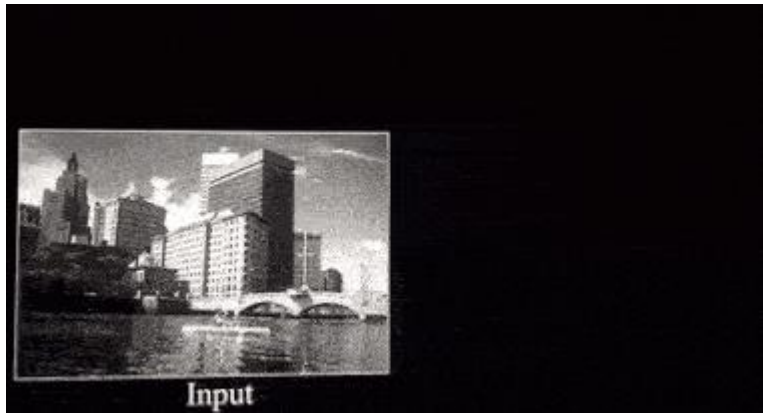
Convolution Layer (Filter)

Convolution Layer (Filter)

How is it helping?

*

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

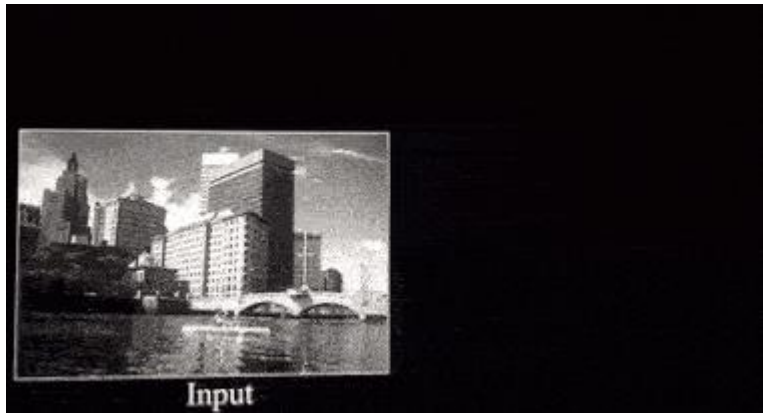


Convolution Layer (Filter)

How is it helping?

$*$

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



Let's do an Example

1	3	1
0	-1	1
2	2	-1

input

$*$

1	2
0	-1

kernel

flip(kernel)

-1	0
2	1

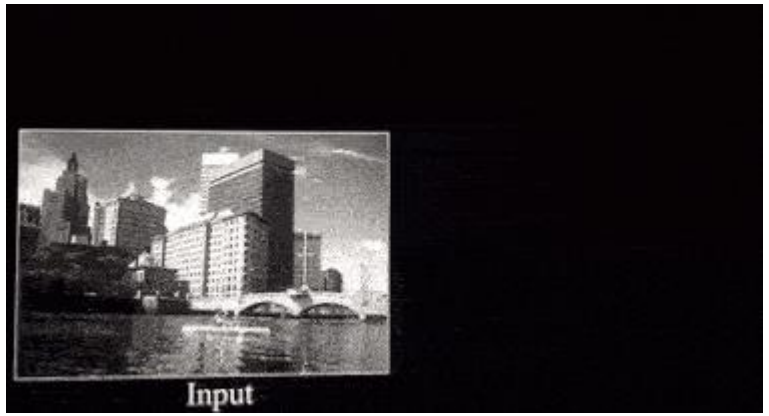
Multiply the window element by element with the flip(kernel), then sum the results

Convolution Layer (Filter)

How is it helping?

$*$

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



Let's do an Example

1	3	1
0	-1	1
2	2	-1

input

$*$

1	2
0	-1

kernel

flip(kernel)

-1	0
2	1

Multiply the window element by element with the flip(kernel), then sum the results

1	**3**	1
0	**-1**	1
2	2	-1

=> 1(-1)+3(0)+0(2)-1(1) = -2

1	**3**	**1**
0	**-1**	**1**
2	2	-1

=> 3(-1)+1(0)-1(2)+1(1) = -4

1	3	1
0	**-1**	1
2	**2**	-1

=> 0(-1)-1(0)+2(2)+2(1) = 6

1	3	1
0	**-1**	**1**
2	**2**	**-1**

=> -1(-1)+1(0)+2(2)-1(1) = 4

result(valid)

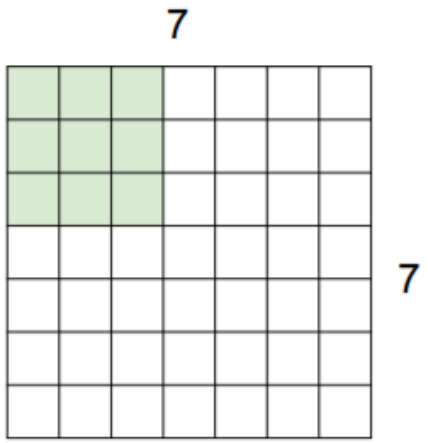
-2	-4
6	4

Convolution Layer

- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***

Convolution Layer

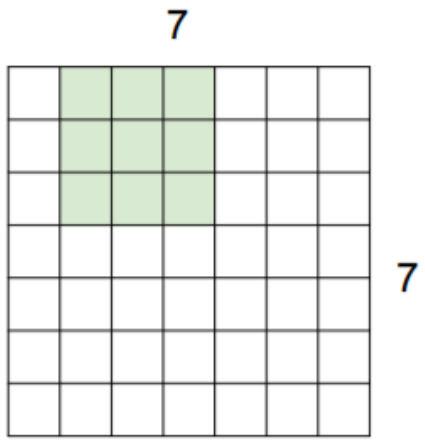
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***



Stride : 1

Convolution Layer

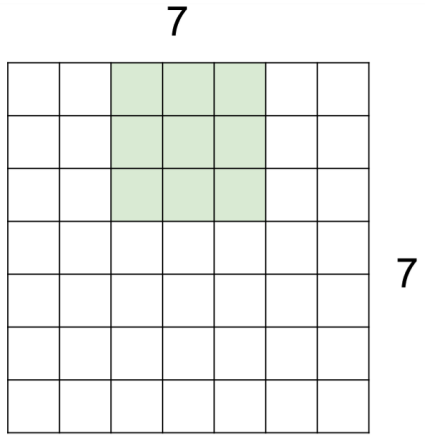
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***



Stride : 1

Convolution Layer

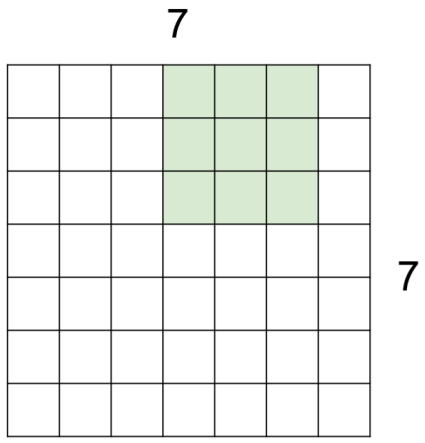
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***



Stride : 1

Convolution Layer

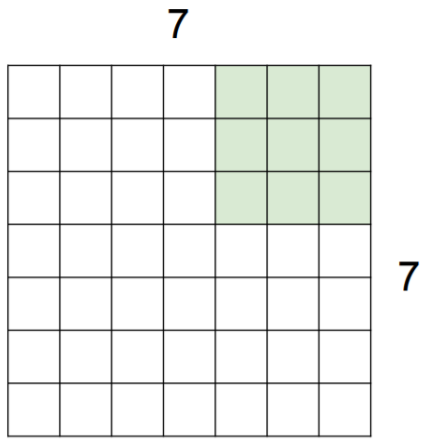
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***



Stride : 1

Convolution Layer

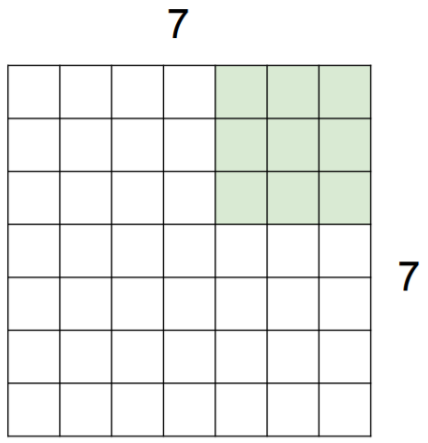
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***



Stride : 1

Convolution Layer

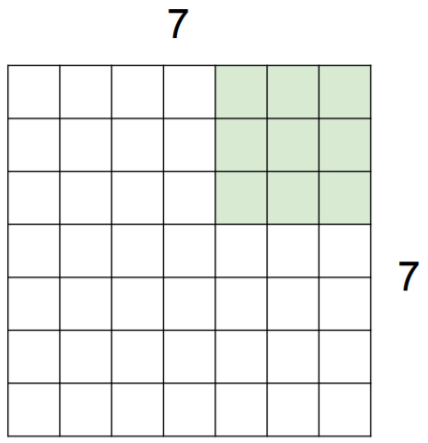
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***
- Do you think we can change the strides?



Stride : 1

Convolution Layer

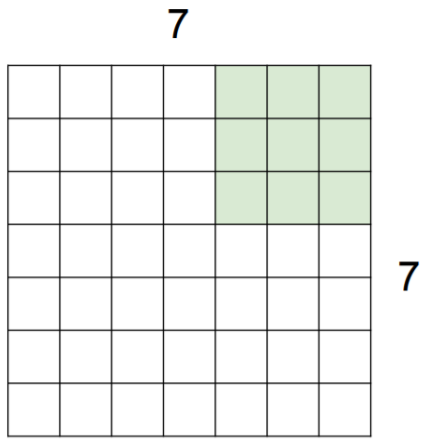
- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***
- Do you think we can change the strides? **Yes!**



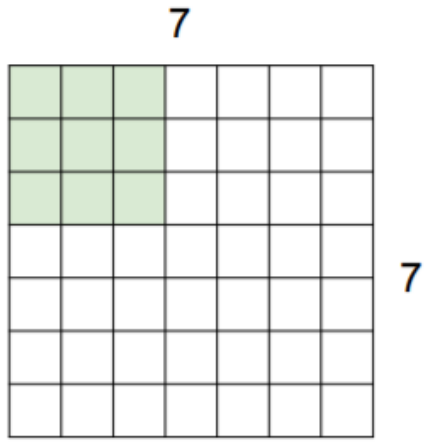
Stride : 1

Convolution Layer

- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***
- Do you think we can change the strides? **Yes!**



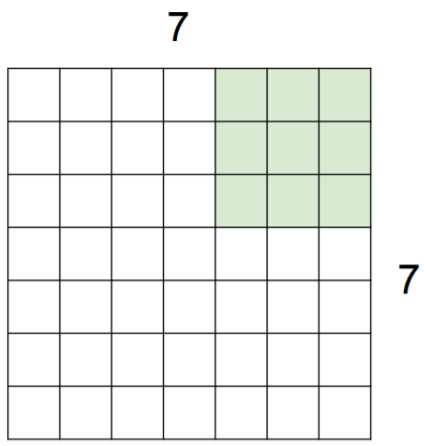
Stride : 1



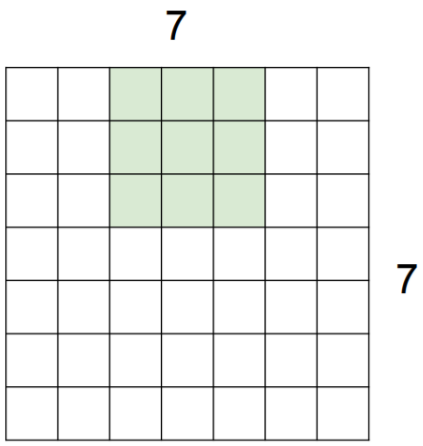
Stride : 2

Convolution Layer

- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***
- Do you think we can change the strides? **Yes!**



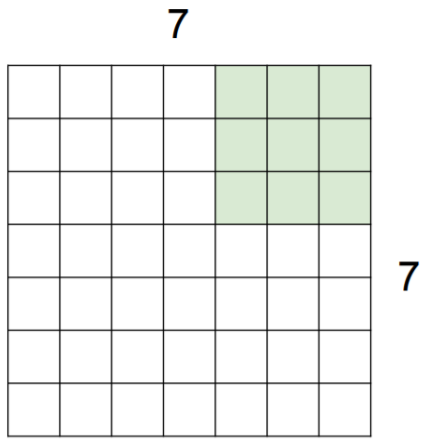
Stride : 1



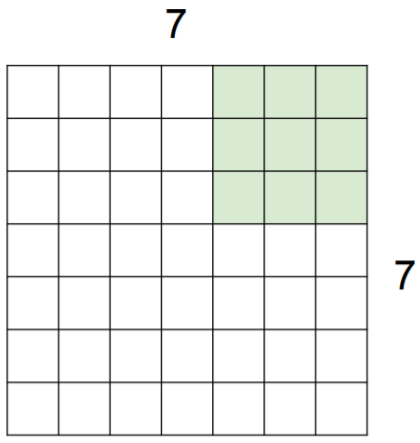
Stride : 2

Convolution Layer

- While convoluting so far, we have shown a sliding window of 1. This is called a ***STRIDE.***
- Do you think we can change the strides? **Yes!**



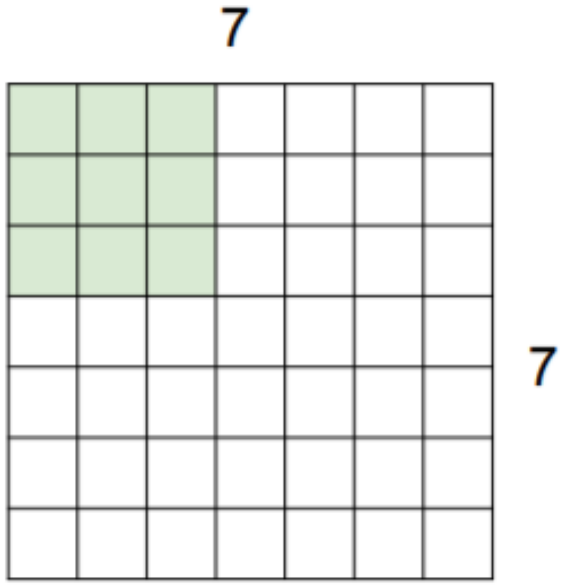
Stride : 1



Stride : 2

Convolution Layer

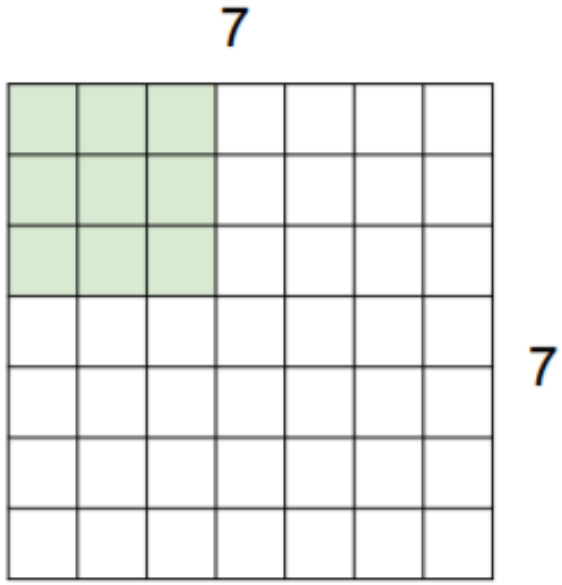
- What is the dimension of the transformed image?



Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 1

Convolution Layer

- What is the dimension of the transformed image?

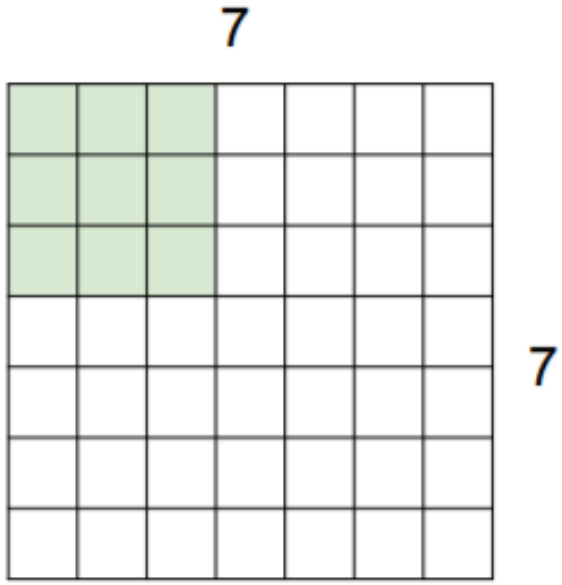


Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 1

Output Dimension: $\frac{N-F}{Stride} + 1$

Convolution Layer

- What is the dimension of the transformed image?

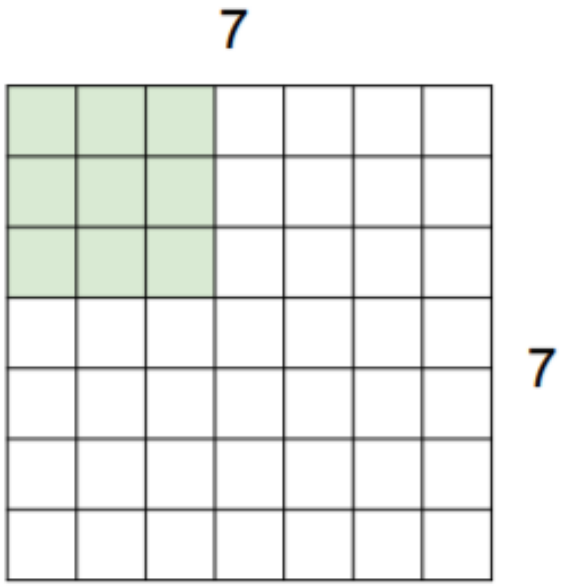


Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 1
Output Dimension: 5x5 (output)

Output Dimension: $\frac{N-F}{Stride} + 1$

Convolution Layer

- What is the dimension of the transformed image?

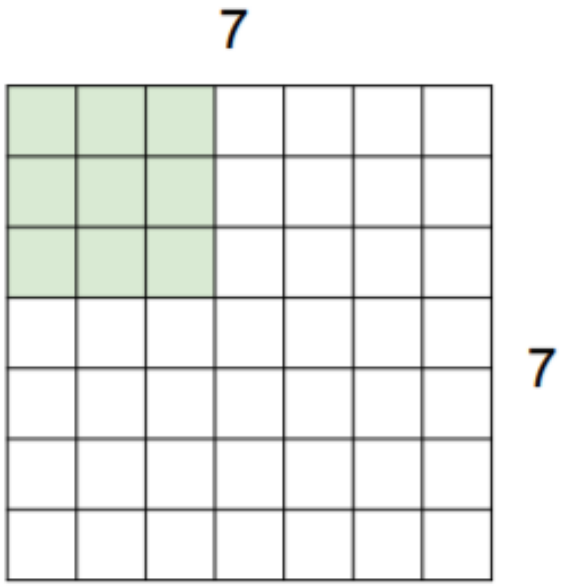


Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 2

Output Dimension: $\frac{N-F}{Stride} + 1$

Convolution Layer

- What is the dimension of the transformed image?

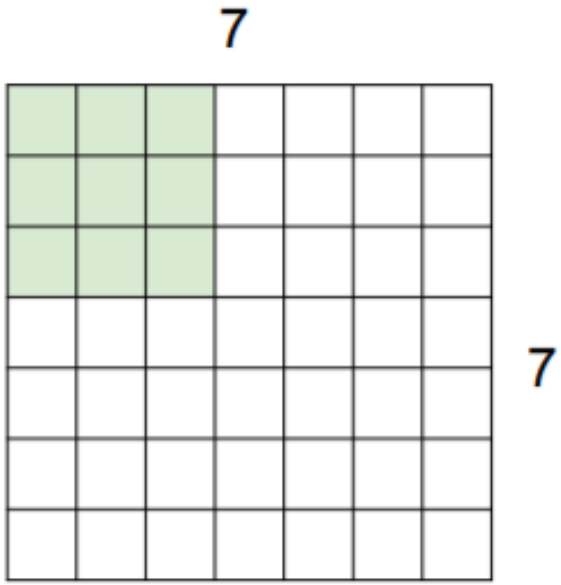


Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 2
Output Dimension: 3x3 (output)

Output Dimension: $\frac{N-F}{Stride} + 1$

Convolution Layer

- What is the dimension of the transformed image?

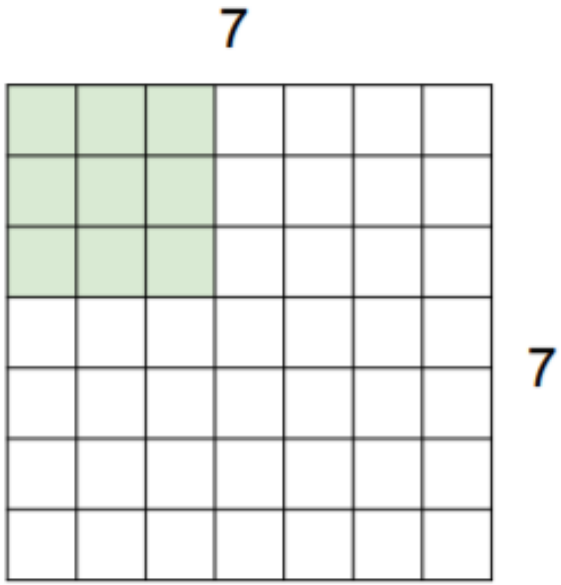


Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 3
Output Dimension: ?

Output Dimension: $\frac{N-F}{Stride} + 1$

Convolution Layer

- What is the dimension of the transformed image?



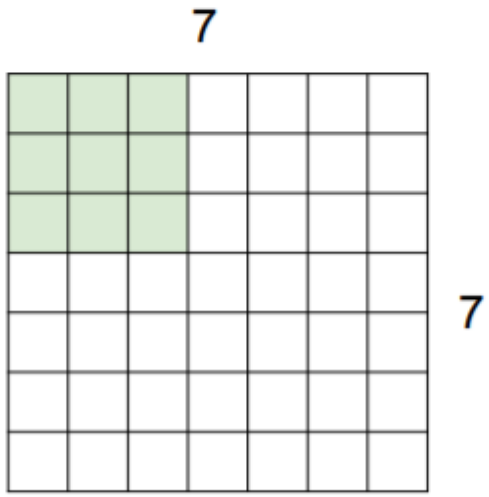
Input Dimension (N): 7x7 (spatially)
Filter Dimension (F) : 3x3 (assume)
Stride : 3
Output Dimension: ?

Output Dimension: $\frac{N-F}{Stride} + 1$

Doesn't fit!

Convolution Layer

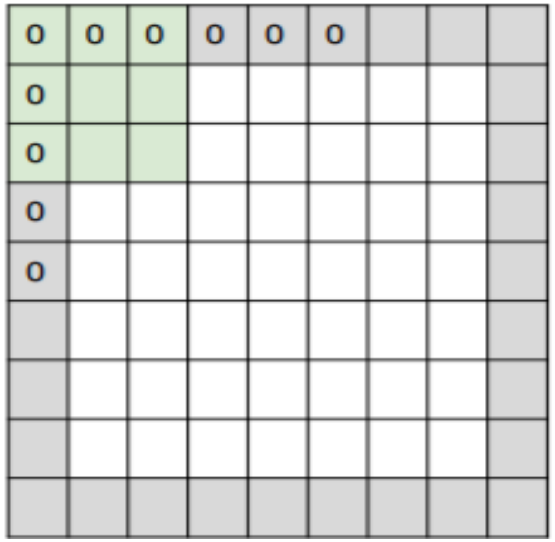
- What is the dimension of the transformed image?



Convolution Layer

- What is the dimension of the transformed image?

In practice: Common to zero pad (Z) the border!

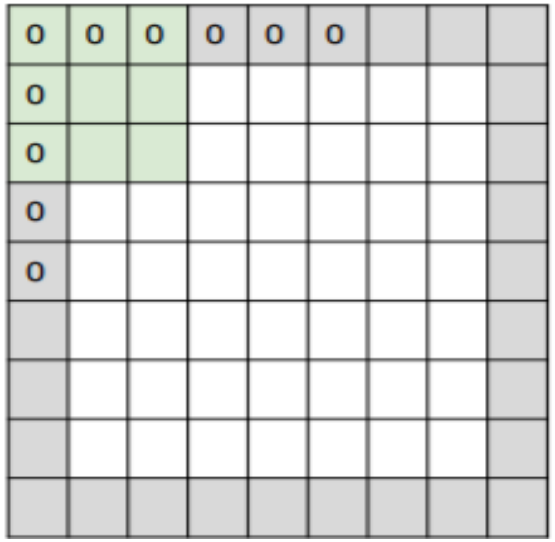


Pad with 1 pixel

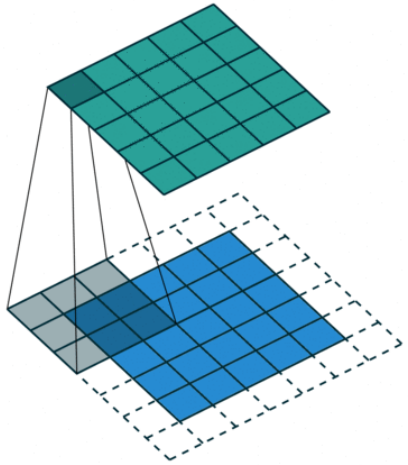
Convolution Layer

- What is the dimension of the transformed image?

In practice: Common to zero pad (Z) the border!



Pad with 1 pixel

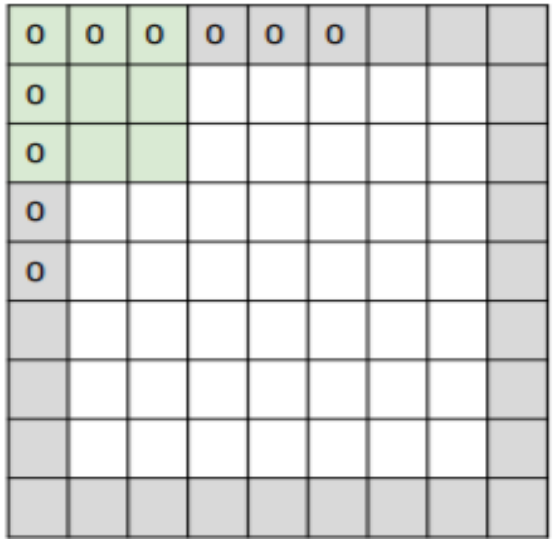


Padding (P): 1

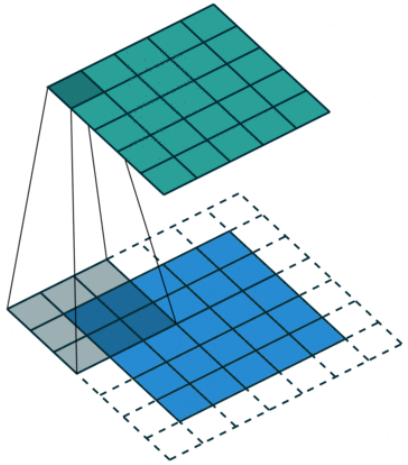
Convolution Layer

- What is the dimension of the transformed image?

In practice: Common to zero pad (Z) the border!



Pad with 1 pixel

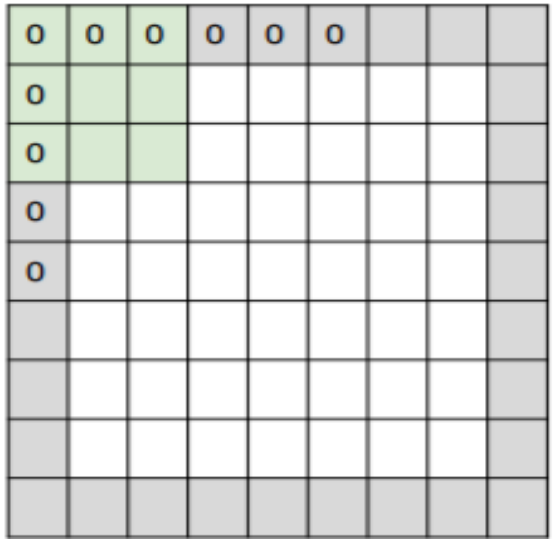


Padding (P): 1

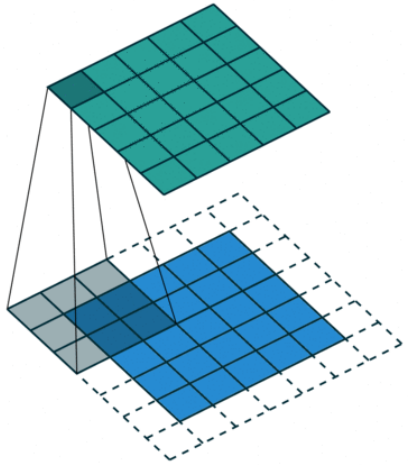
Convolution Layer

- What is the dimension of the transformed image?

In practice: Common to zero pad (Z) the border!



Pad with 1 pixel



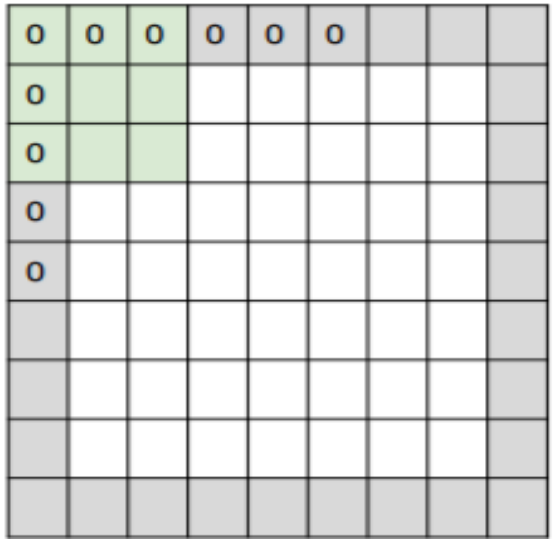
Output Dimension: $\frac{N-F+2P}{Stride}+1$

Padding (P): 1

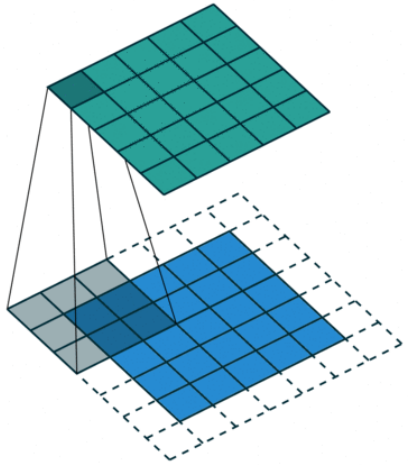
Convolution Layer

- What is the dimension of the transformed image?

In practice: Common to zero pad (Z) the border!



Pad with 1 pixel



Output Dimension: $\frac{N-F+2P}{Stride}+1$

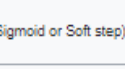
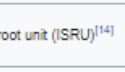
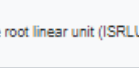
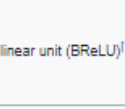
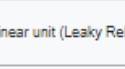
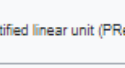
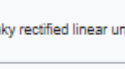
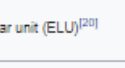
Padding (P): 1
Output Dimension: 3 x 3

Convolution Layer

- In summary, we have ***four*** important hyper-parameters:
  1. Number of Filters, K
  2. Size of the Filter, F
  3. Stride, S
  4. Zero Padding, Z

Activation Function

- A value passed through the function will be transformed within a range.

Name	Plot	Equation
Identity		
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$ [1]
TanH		$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$
SQNL[9]		$f(x) = \begin{cases} 1 & : x > 2.0 \\ x - \frac{x^2}{4} & : 0 \leq x \leq 2.0 \\ x + \frac{x^2}{4} & : -2.0 \leq x < 0 \\ -1 & : x < -2.0 \end{cases}$
ArcTan		$f(x) = \tan^{-1}(x)$
ArSinH		$f(x) = \sinh^{-1}(x) = \ln\left(x + \sqrt{x^2 + 1}\right)$
ElliotSig[10][11] Softsign[12][13]		$f(x) = \frac{x}{1 + \lvert x \rvert}$
Inverse square root unit (ISRU)[14]		$f(x) = \frac{x}{\sqrt{1 + \alpha x^2}}$
Inverse square root linear unit (ISRLU)[14]		$f(x) = \begin{cases} \frac{x}{\sqrt{1+\alpha x^2}} & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Rectified linear unit (ReLU)[15]		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$
Bipolar rectified linear unit (BReLU)[16]		$f(x_i) = \begin{cases} ReLU(x_i) & \text{if } i \bmod 2 = 0 \\ -ReLU(-x_i) & \text{if } i \bmod 2 \neq 0 \end{cases}$
Leaky rectified linear unit (Leaky ReLU)[17]		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Parameteric rectified linear unit (PReLU)[18]		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Randomized leaky rectified linear unit (RReLU)[19]		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \text{ [3]} \\ x & \text{for } x \geq 0 \end{cases}$
Exponential linear unit (ELU)[20]		$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$
Scaled exponential linear unit (SELU)[21]		$f(\alpha, x) = \lambda \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ with $\lambda = 1.0507$ and $\alpha = 1.67326$
S-shaped rectified linear activation unit (SReLU)[22]		$f_{t_l, a_l, t_r, a_r}(x) = \begin{cases} t_l + a_l(x - t_l) & \text{for } x \leq t_l \\ x & \text{for } t_l < x < t_r \\ t_r + a_r(x - t_r) & \text{for } x \geq t_r \end{cases}$ t_l, a_l, t_r, a_r are parameters.
Adaptive piecewise linear (APL)[23]		$f(x) = \max(0, x) + \sum_{s=1}^{S} a_i^s \max(0, -x + b_i^s)$

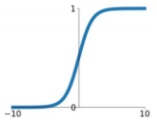
https://en.wikipedia.org/wiki/Activation_function

Activation Function

- A value passed through the function will be transformed within a range.

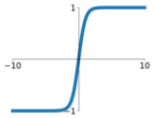
Sigmoid

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



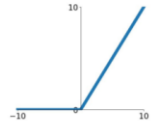
tanh

$\tanh(x)$



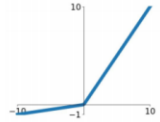
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

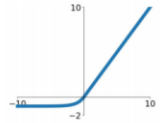


Maxout

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

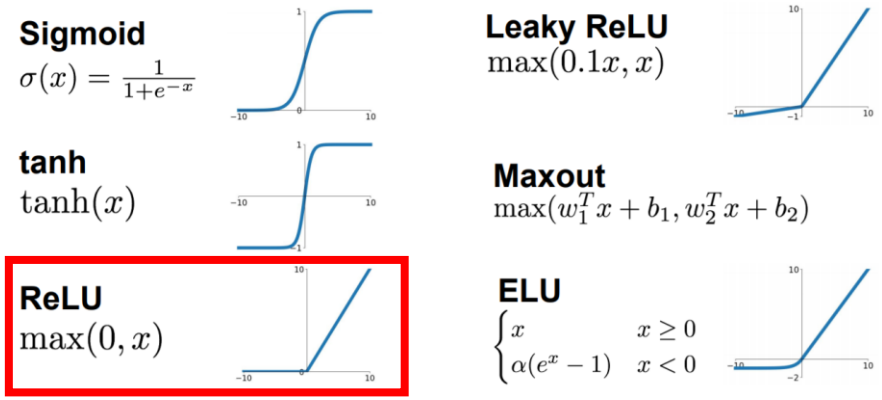
ELU

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



Activation Function

- A value passed through the function will be transformed within a range.
- The most used activation function in CNNs is the ReLU (Rectified Linear Unit).



Activation Function

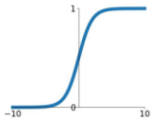
- A value passed through the function will be transformed within a range.
- The most used activation function in CNNs is the ReLU (Rectified Linear Unit).

1	14	-9	4
-2	-20	10	6
-3	3	11	1
2	54	-2	80

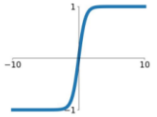
ReLU

1	14	0	4
0	0	10	6
0	3	11	1
2	54	0	80

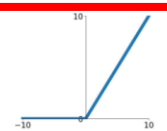
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



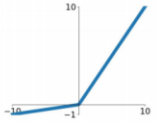
tanh
$\tanh(x)$



ReLU
$\max(0, x)$

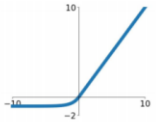


Leaky ReLU
$\max(0.1x, x)$



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

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



Activation Function

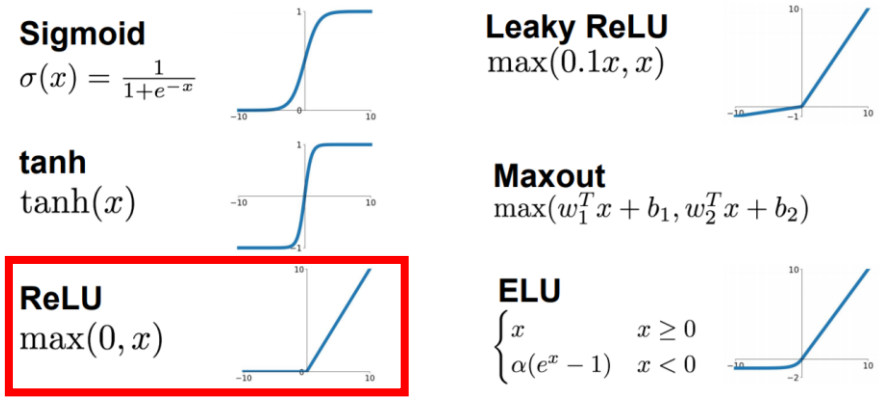
- A value passed through the function will be transformed within a range.
- The most used activation function in CNNs is the ReLU (Rectified Linear Unit).

1	14	-9	4
-2	-20	10	6
-3	3	11	1
2	54	-2	80

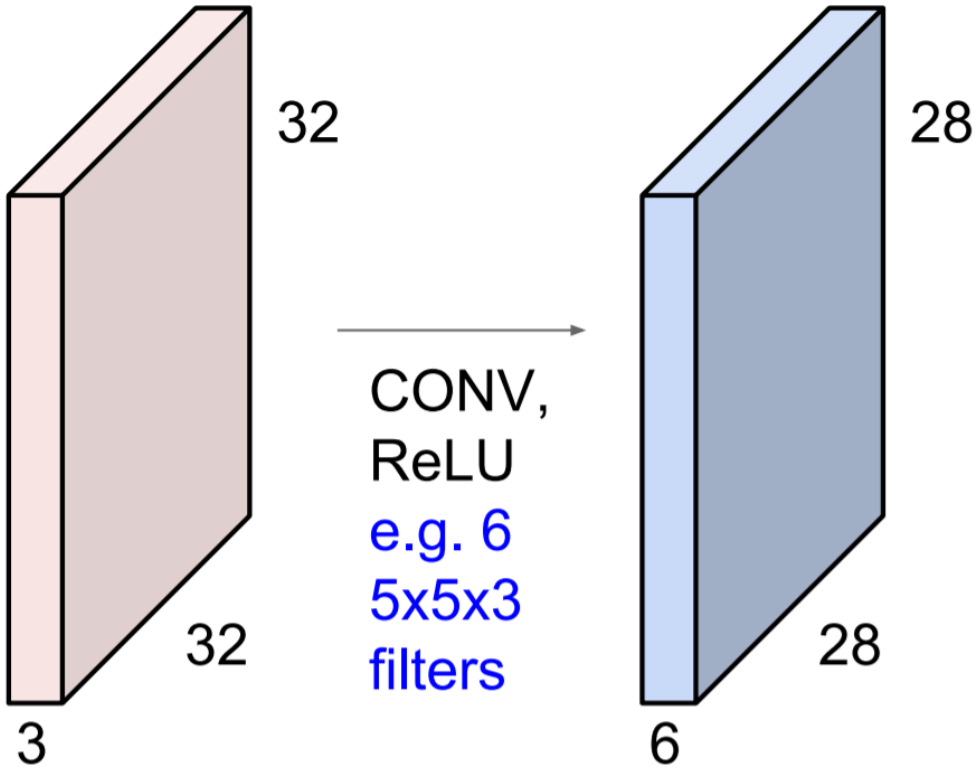
ReLU

1	14	0	4
0	0	10	6
0	3	11	1
2	54	0	80

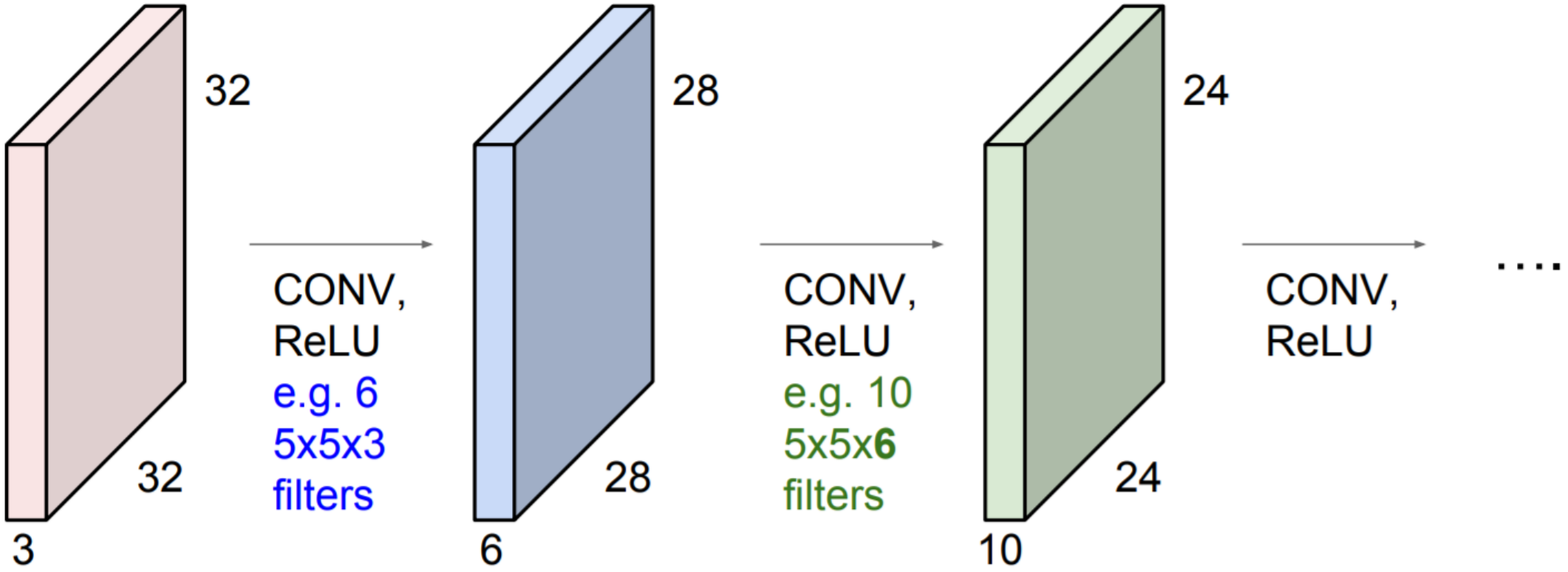
- While using **ReLU**, be careful with your selection of **learning rate!**



Convolution + Activation Layer

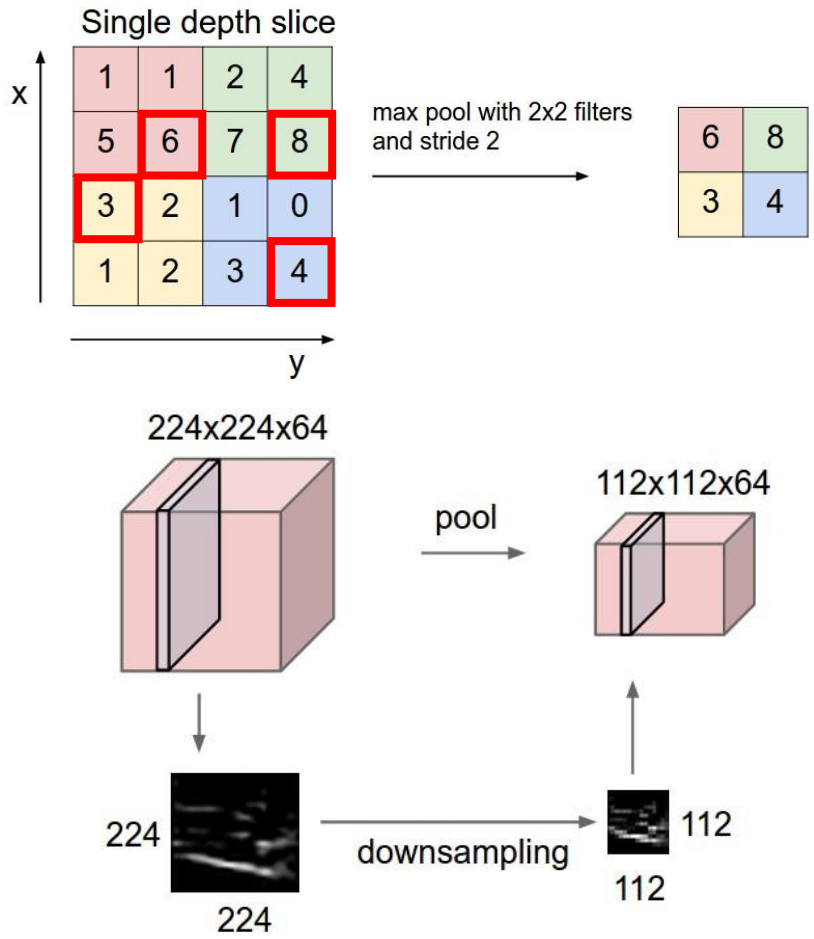


Convolution + Activation Layer



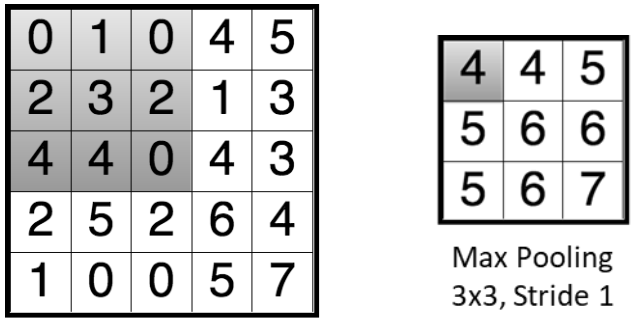
Pooling Layers

- Pooling in a CNN is a subsampling step
  - Replaces the output to summary statistics of nearby outputs!
- It is common to periodically insert a Pooling layer in-between successive Conv layers
- Operates over each activation map independently
- Pooling is performed for two reasons
  1. Dimensionality reduction.
  2. Invariance to transformations of rotation and translation (*If we translate the input by a small amount values of most of the outputs does not change*). This is important *to identify if a feature is present rather than where exactly it is.*



Pooling Layers

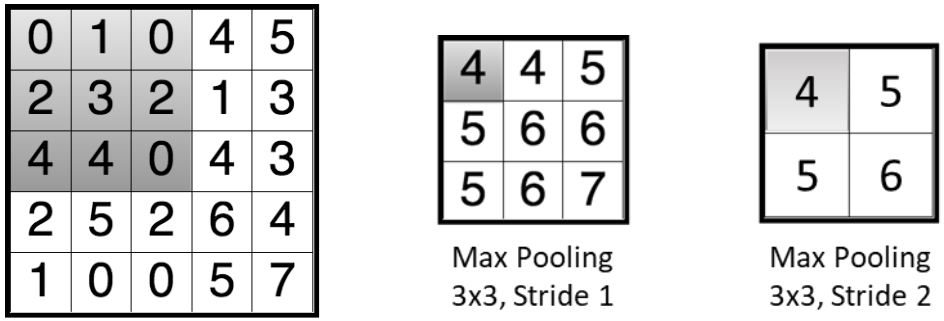
Similar to Convolution layer the strides can be changed!



1. norm of a rectangular neighborhood
2. Weighted average based on the distance from the central pixel

Pooling Layers

Similar to Convolution layer the strides can be changed!



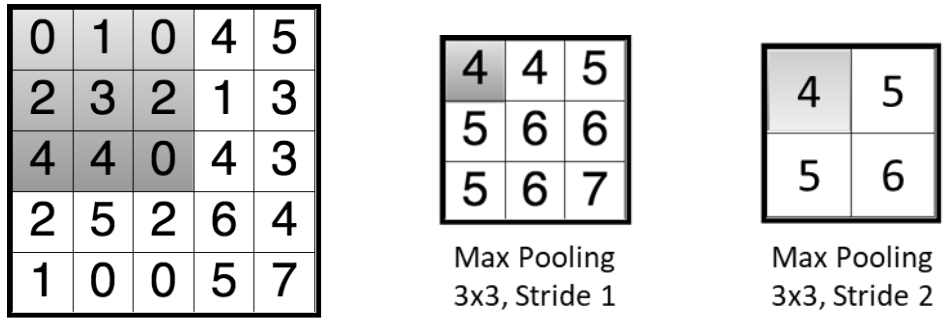
1. norm of a rectangular neighborhood
2. Weighted average based on the distance from the central pixel

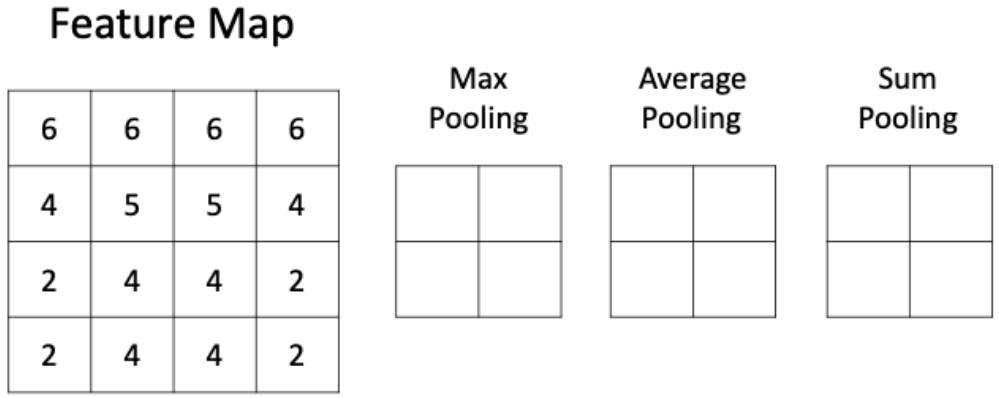
Pooling Layers

$L\,2$ norm of a rectangular neighborhood

Similar to Convolution layer the strides can be changed!

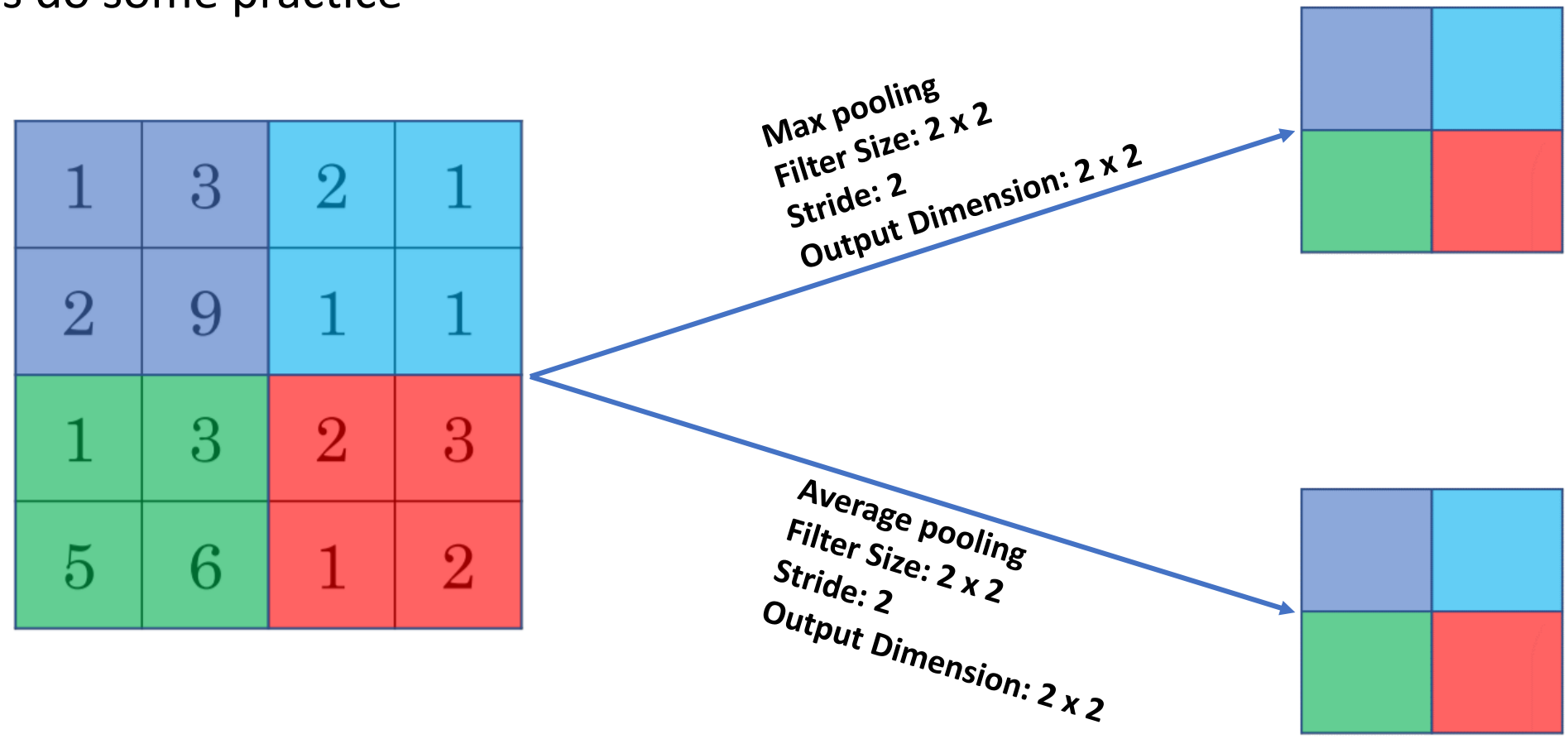
- Popular pooling functions:
  1. Max pooling operation reports the maximum output within a rectangular neighborhood
  2. Average of a rectangular neighborhood
  3. Sum pooling
  4. Weighted average based on the distance from the central pixelWeighted average based on the distance from the central pixel





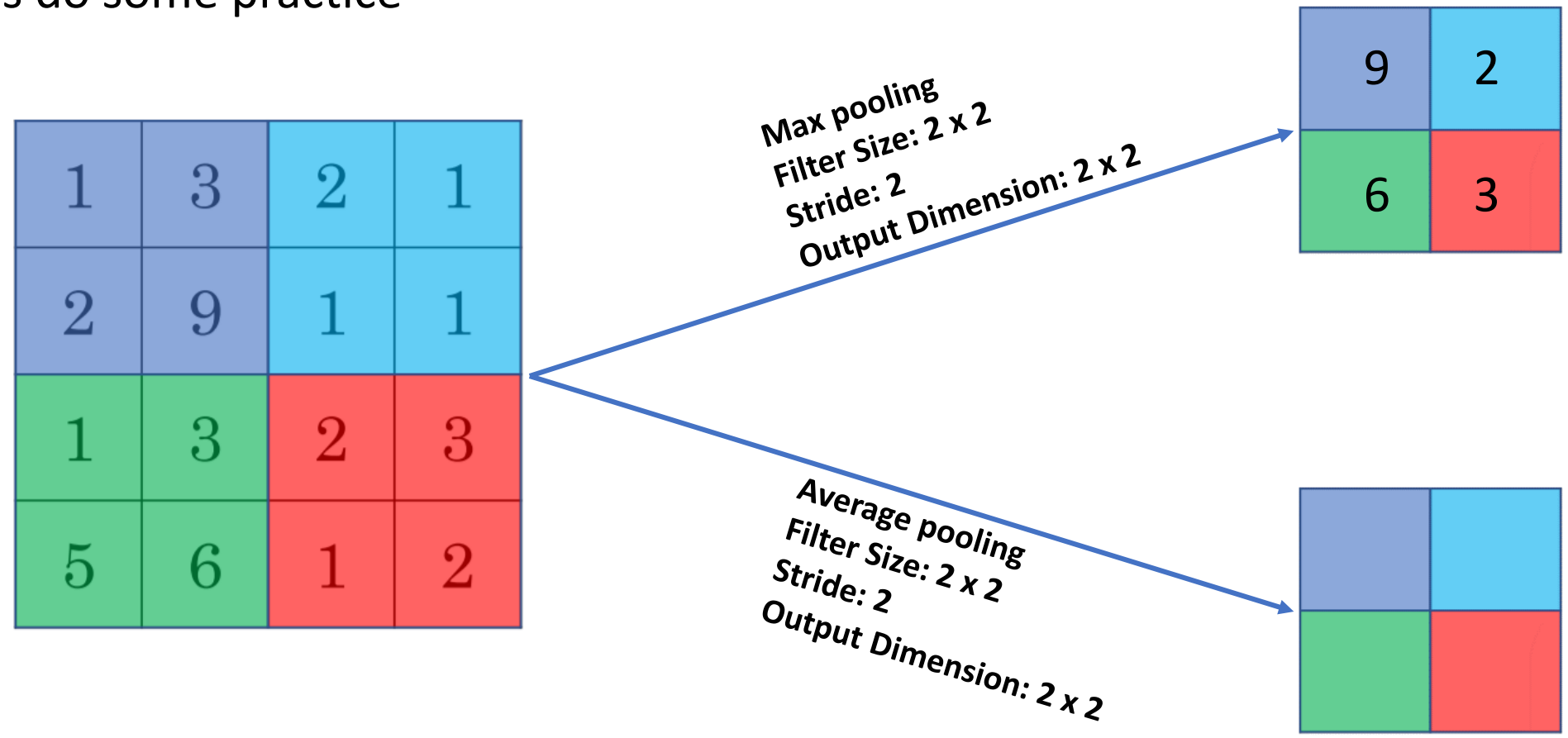
Pooling Layers

- Let's do some practice



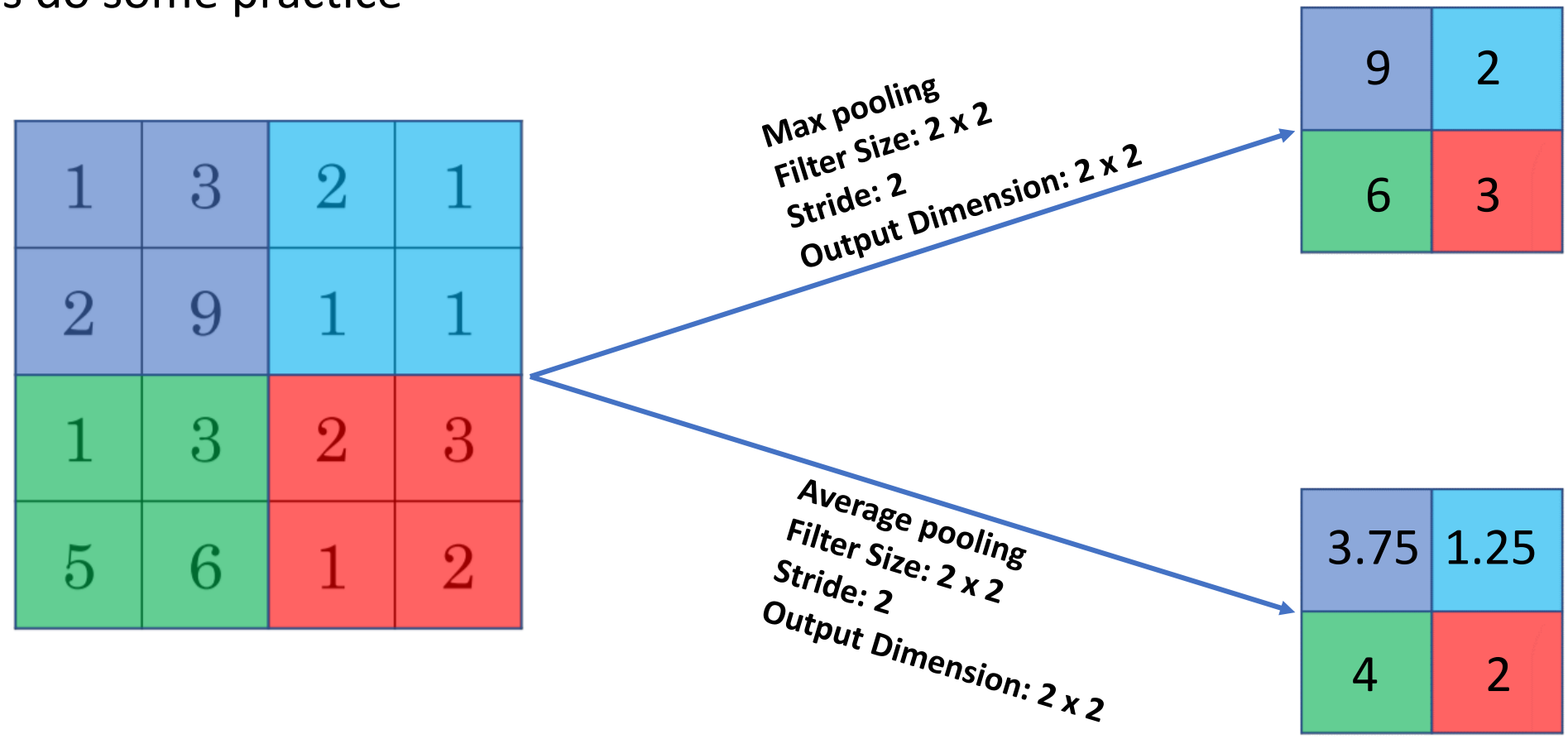
Pooling Layers

- Let's do some practice



Pooling Layers

- Let's do some practice

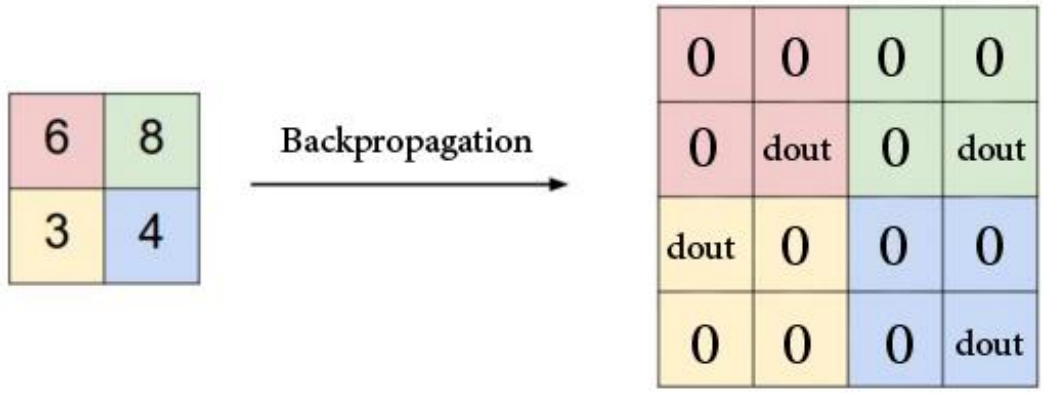


Pooling Layers

- In summary, we have ***three*** important hyper-parameters:
  1. Filters Size, F
  2. Stride, S
  3. Zero Padding are rarely used
- Max pooling or Average pooling popularly are used

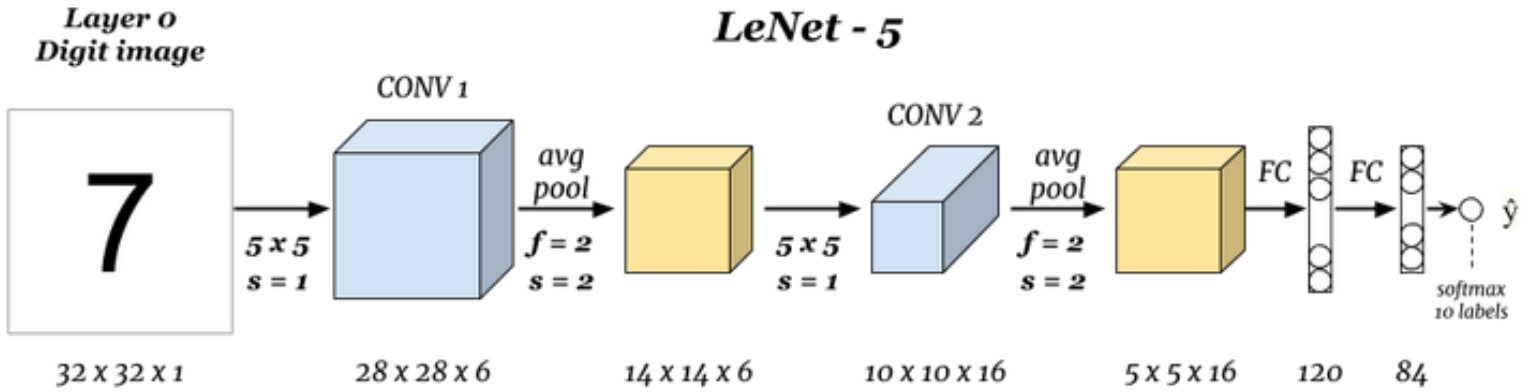
Un-Pooling Layers

- While backpropagating we only preserve the max nodes:



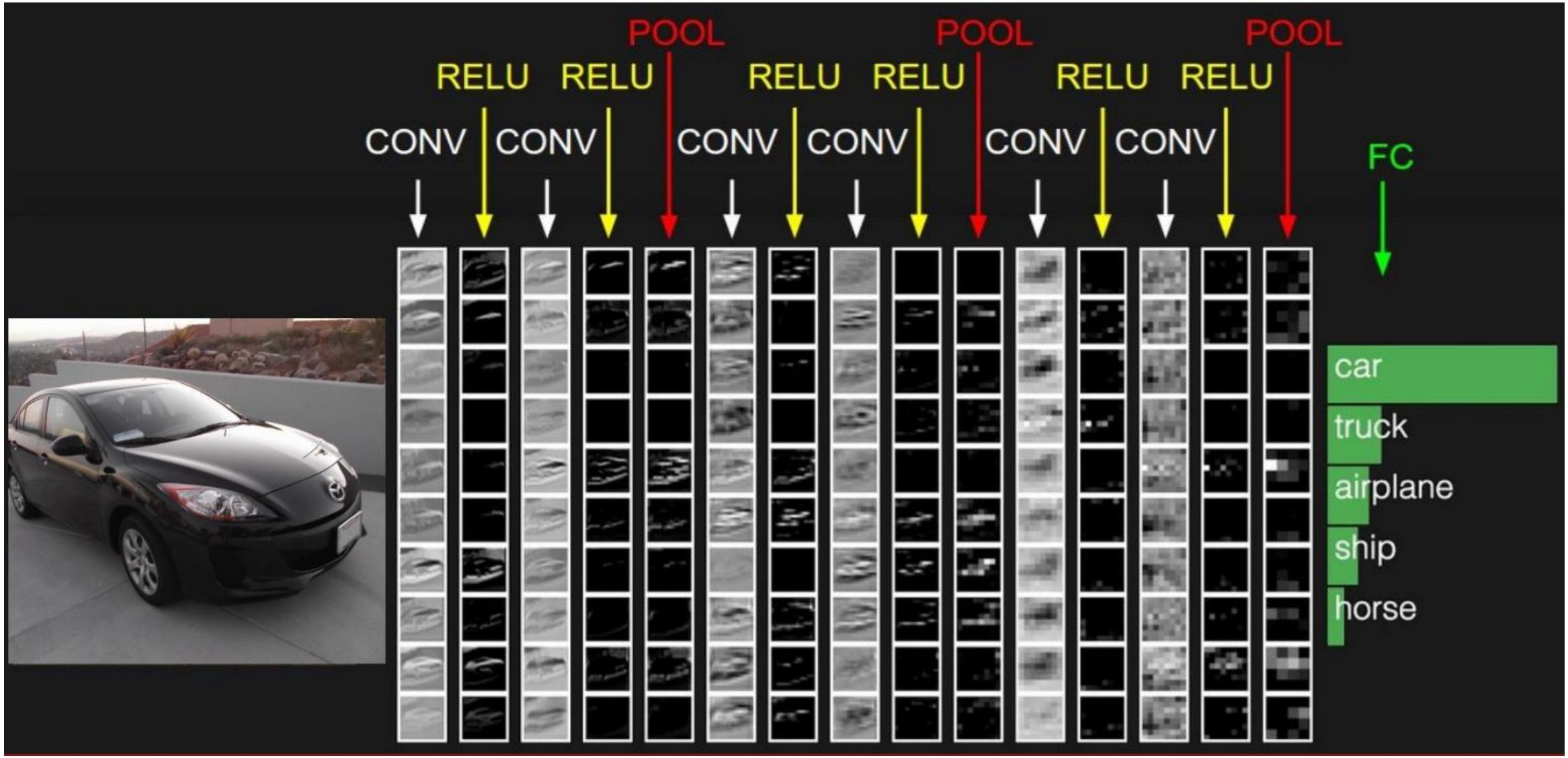
Example 1 (Classic LeNet – 5)

- This model was published in 1998. The last layer wasn't using softmax back then.
- Used for digit recognition.
- The dimensions of the image decreases as the number of channels increases.
- It has *60k* parameters.



LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

Example 2


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse