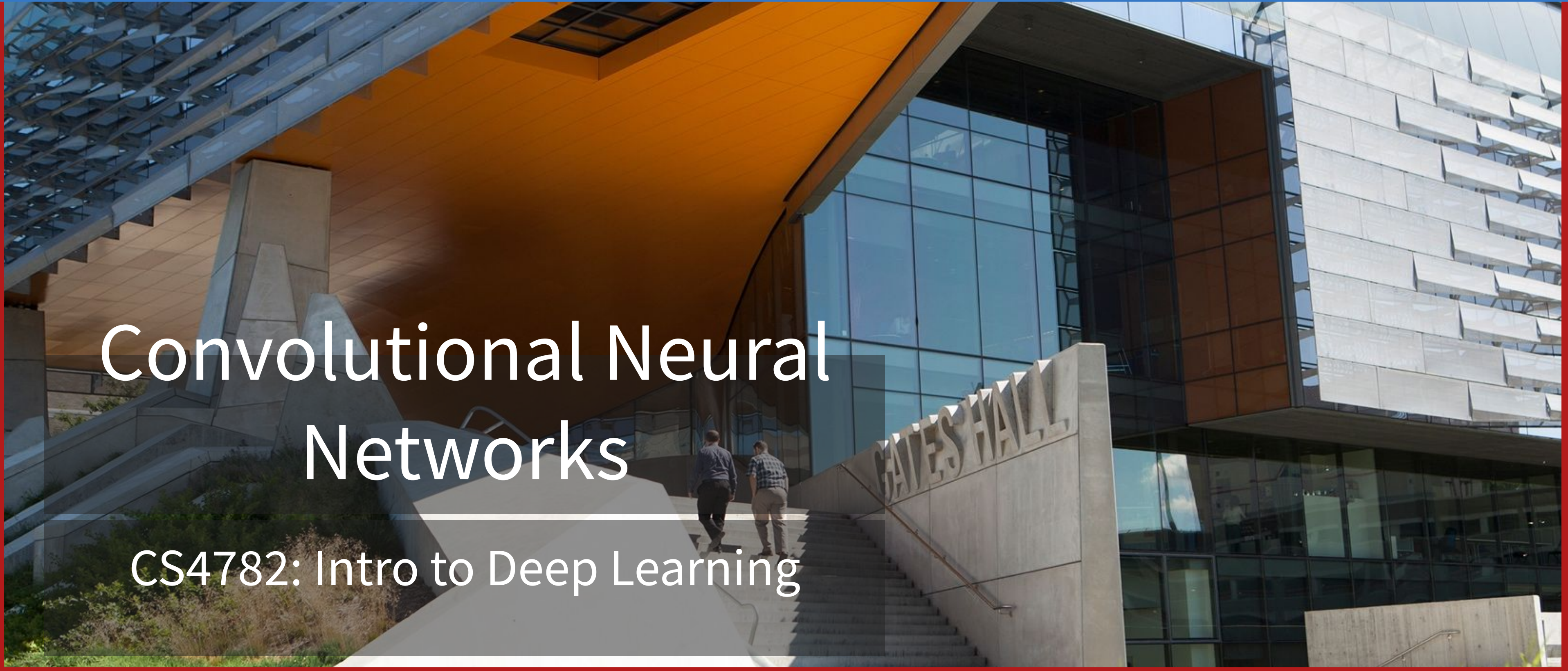
When poll is active respond at **PollEv.com/weichiuma** Send **weichiuma** to **22333**



Convolutional Neural Networks

CS4782: Intro to Deep Learning

Thanks to:

Varsha Kishore
Justin Lovelace
Anissa Dallmann
Stephanie Ginting
Alexander Scotte

Image Classification

input image

classification →

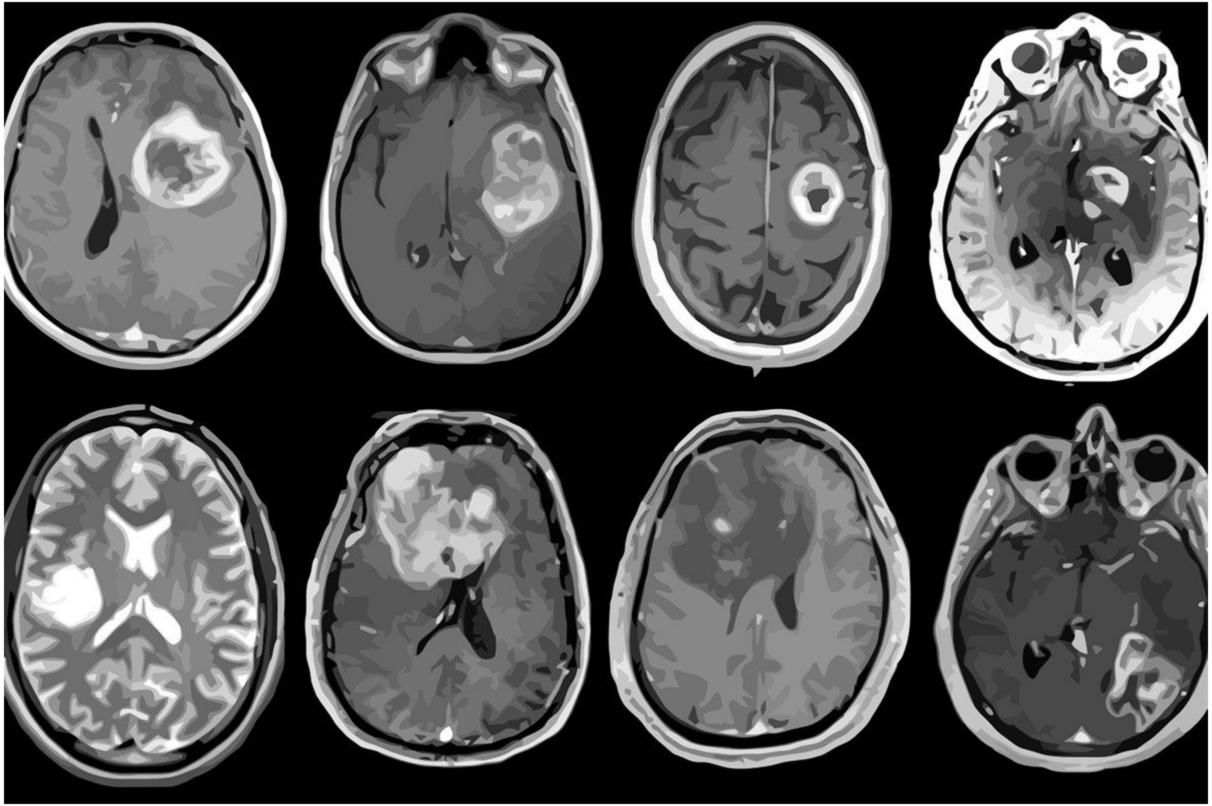
“dog”

input image

classification →

“cat”

Applications in Medicine



Applications in Autonomous Driving

Image Classification

input image

classification →

“dog”

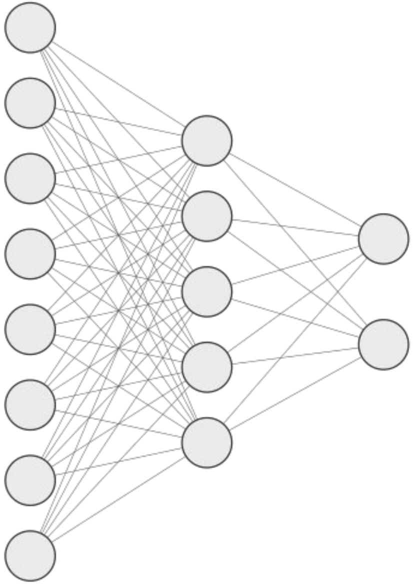
input image

classification →

“cat”

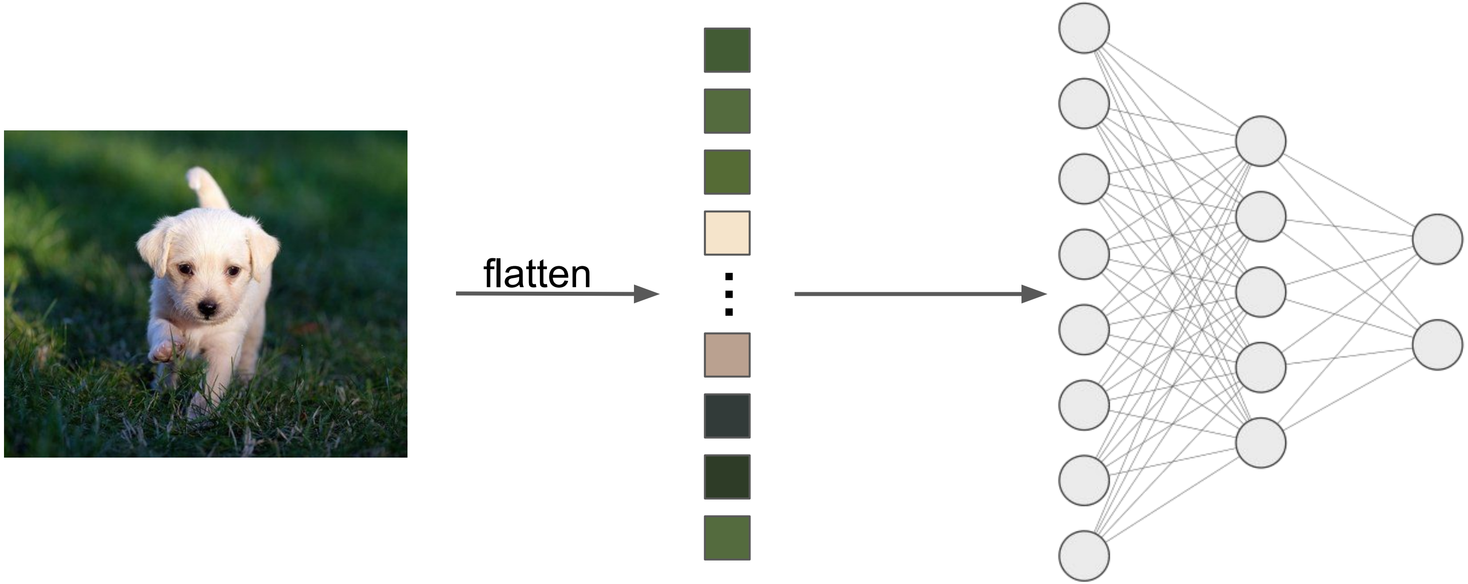
Cornell Bowers C·IS

How to use Multi-Layer Perceptron for image classification?

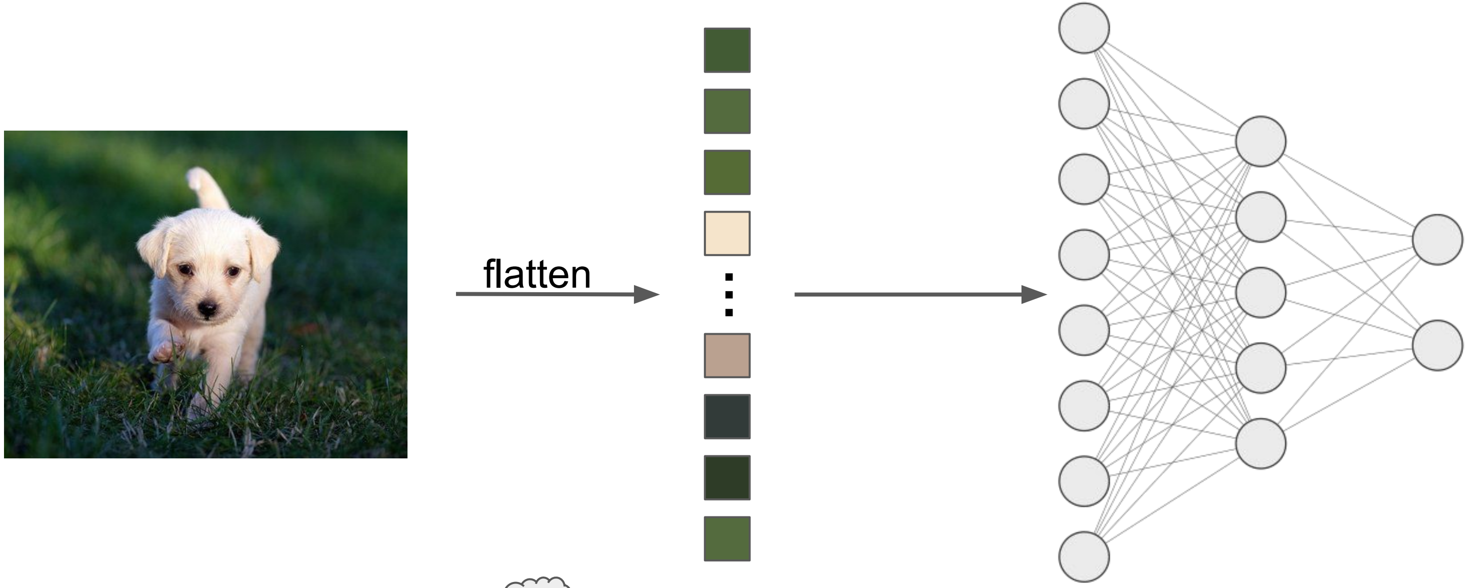


Cornell Bowers C·IS

How to use Multi-Layer Perceptron for image classification?

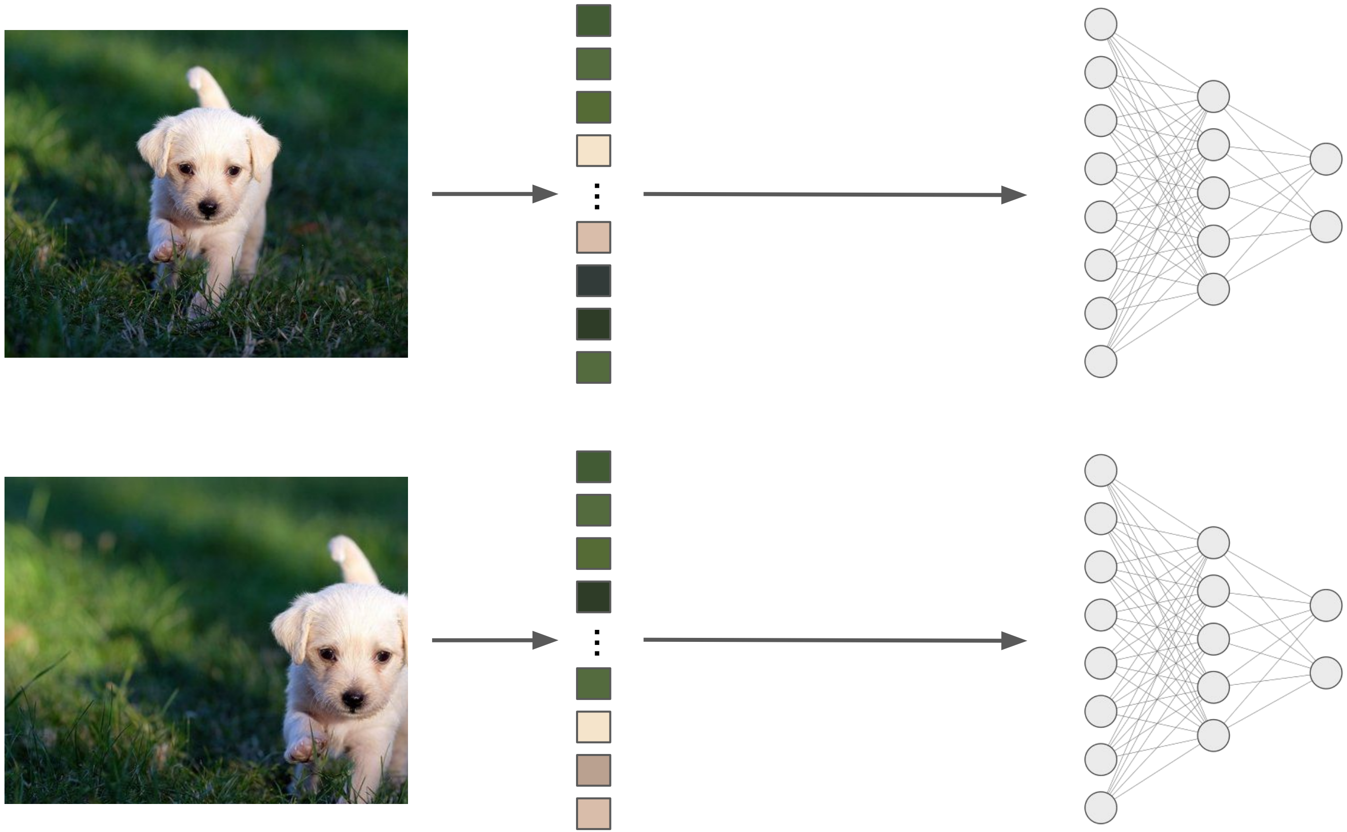


Why not use a Multi-Layer Perceptron?



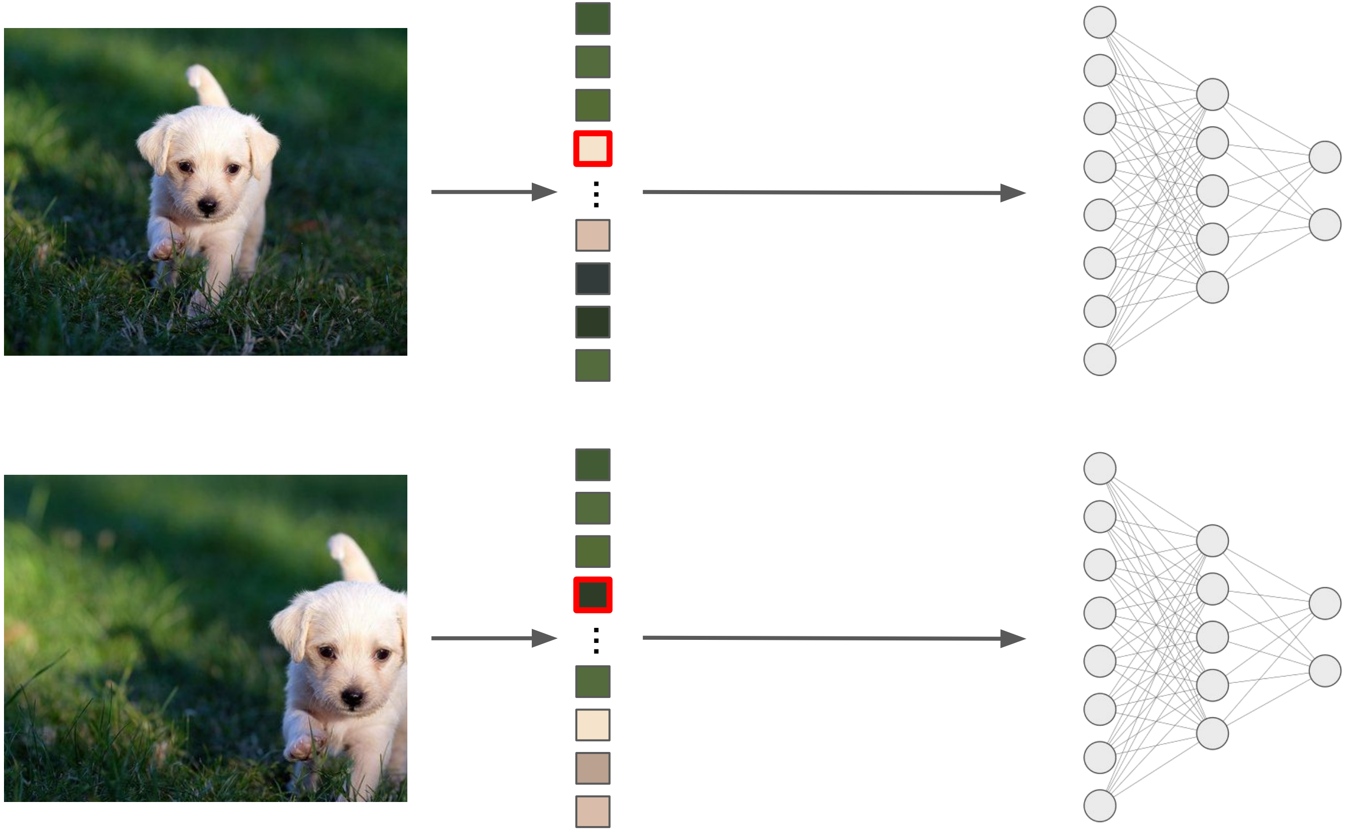
Which pixels were next to each other?

Why not use a Multi-Layer Perceptron?



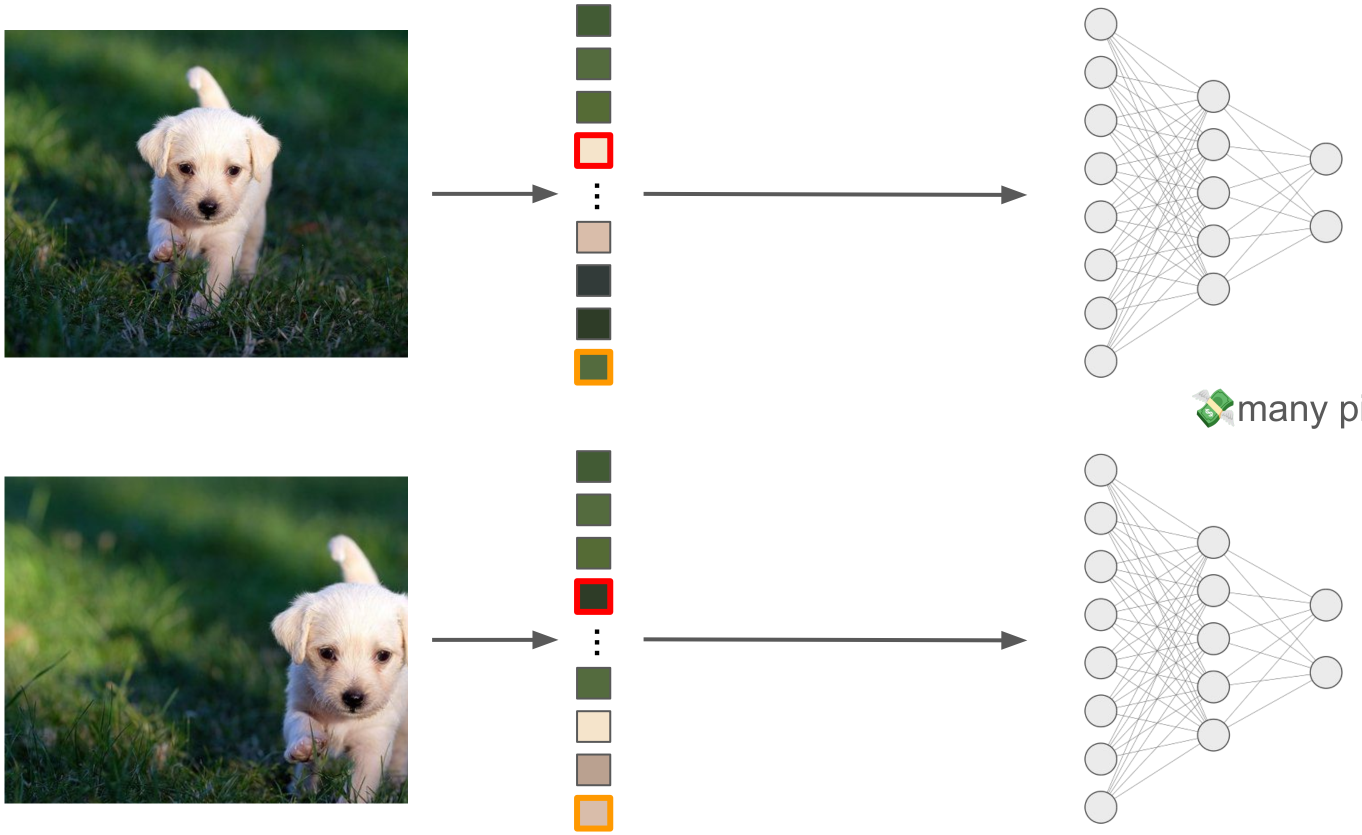
Cornell Bowers C·IS

Why not use a Multi-Layer Perceptron?



Cornell Bowers C·IS

Why not use a Multi-Layer Perceptron?



💸many pixels = many parameters

Cornell Bowers CIS

Our goal today

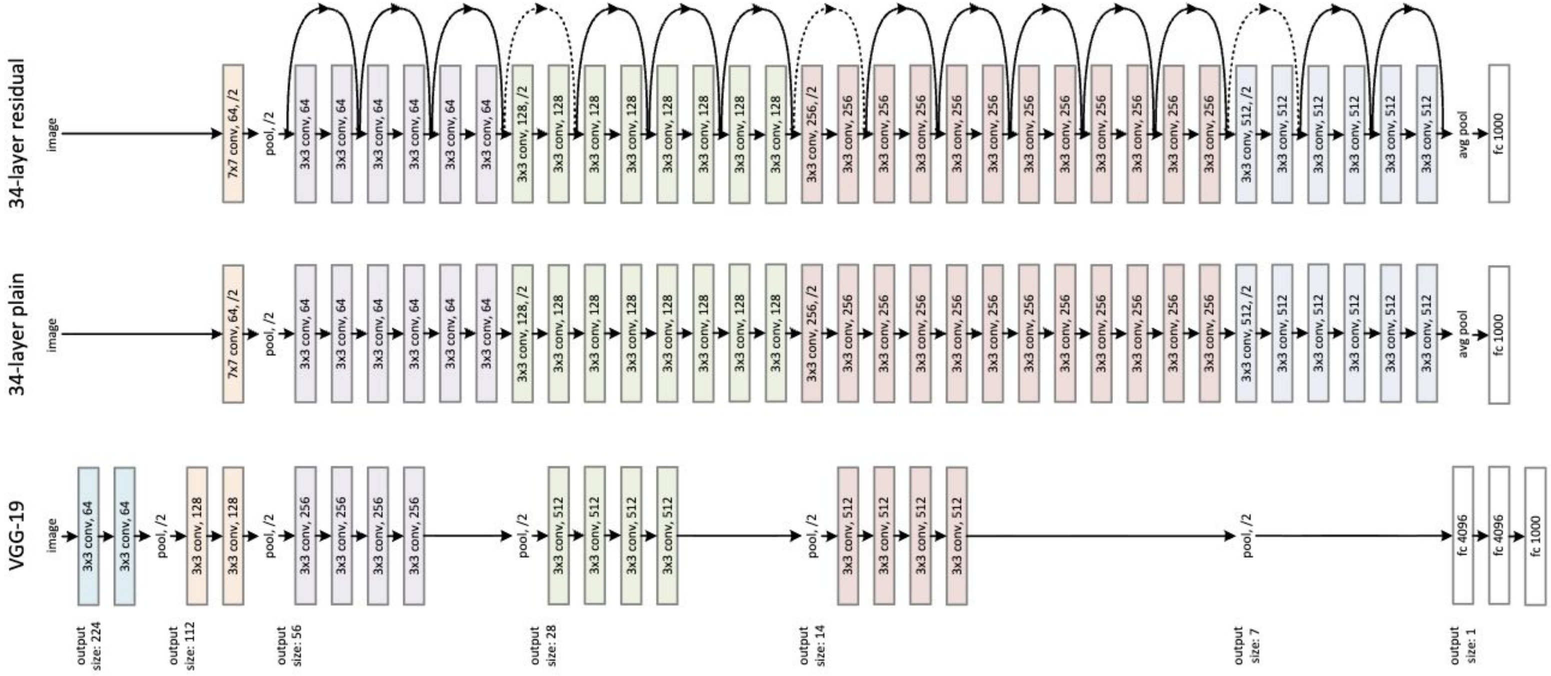
Conv2d

```
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0,
dilation=1, groups=1, bias=True, padding_mode='zeros', device=None, dtype=None)
```

[source]

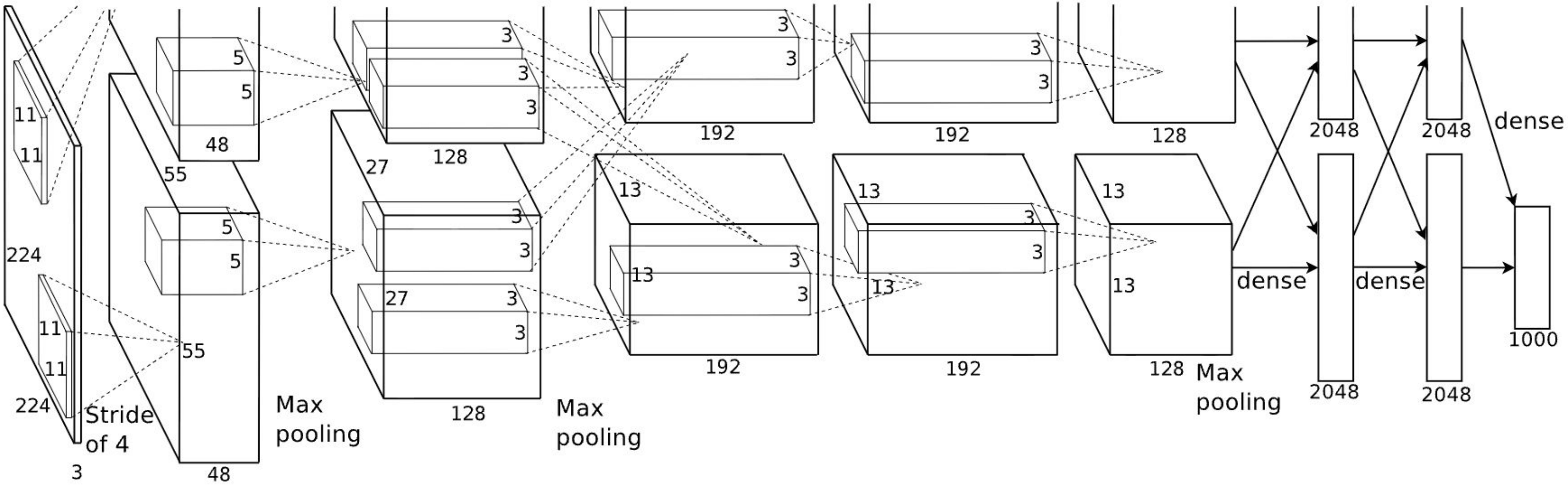
Cornell Bowers C·IS

Our goal today



Cornell Bowers C·IS

Our goal today



Convolutional Filters

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

“image”

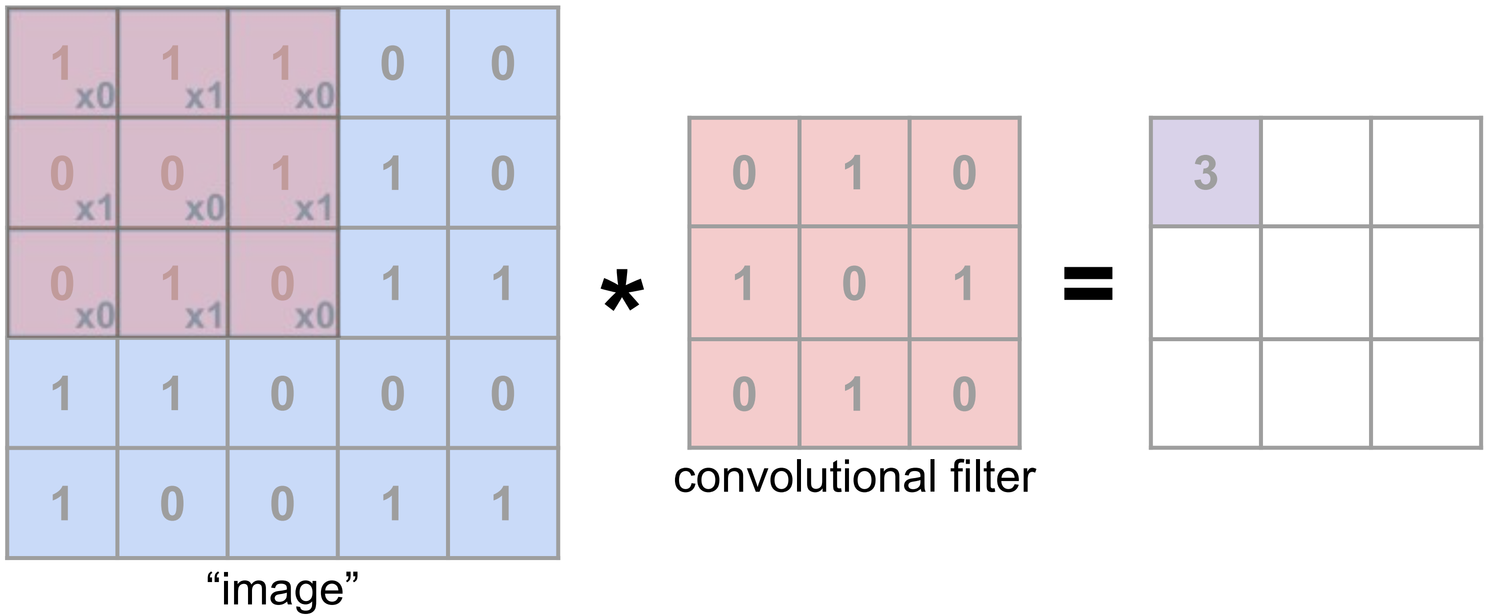
*

0	1	0
1	0	1
0	1	0

convolutional filter

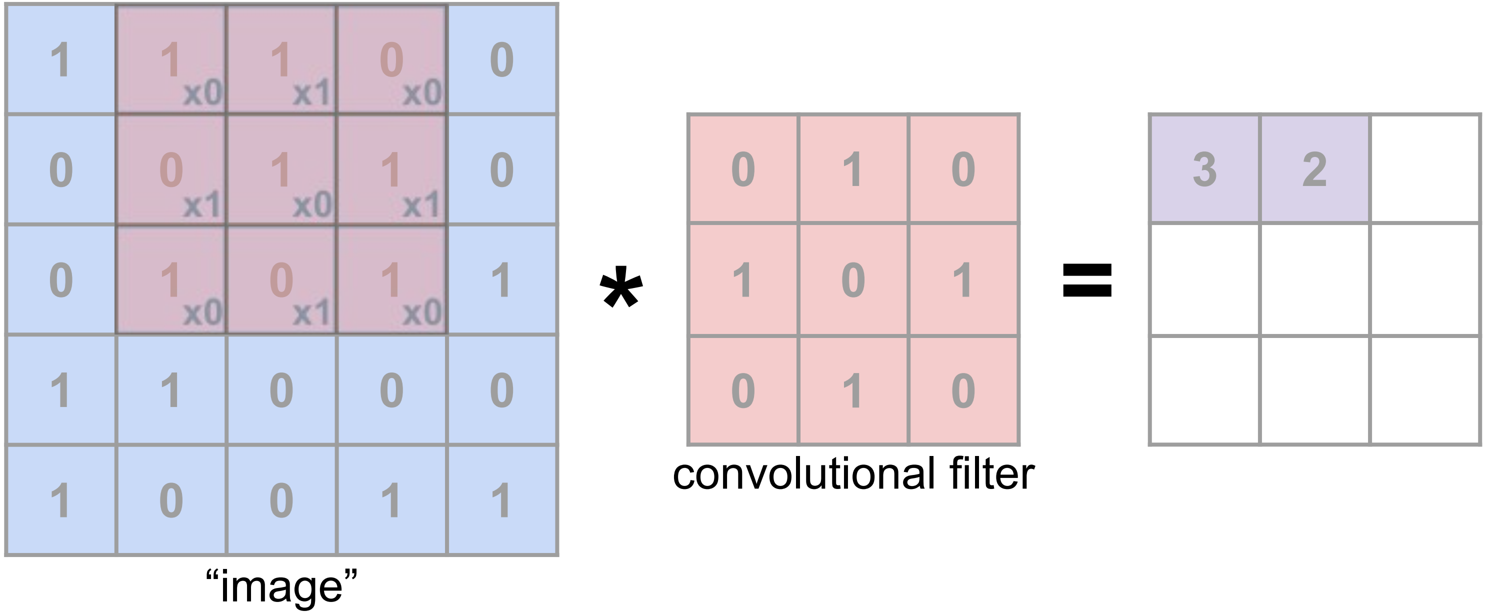
Cornell Bowers C·IS

Convolutional Filters

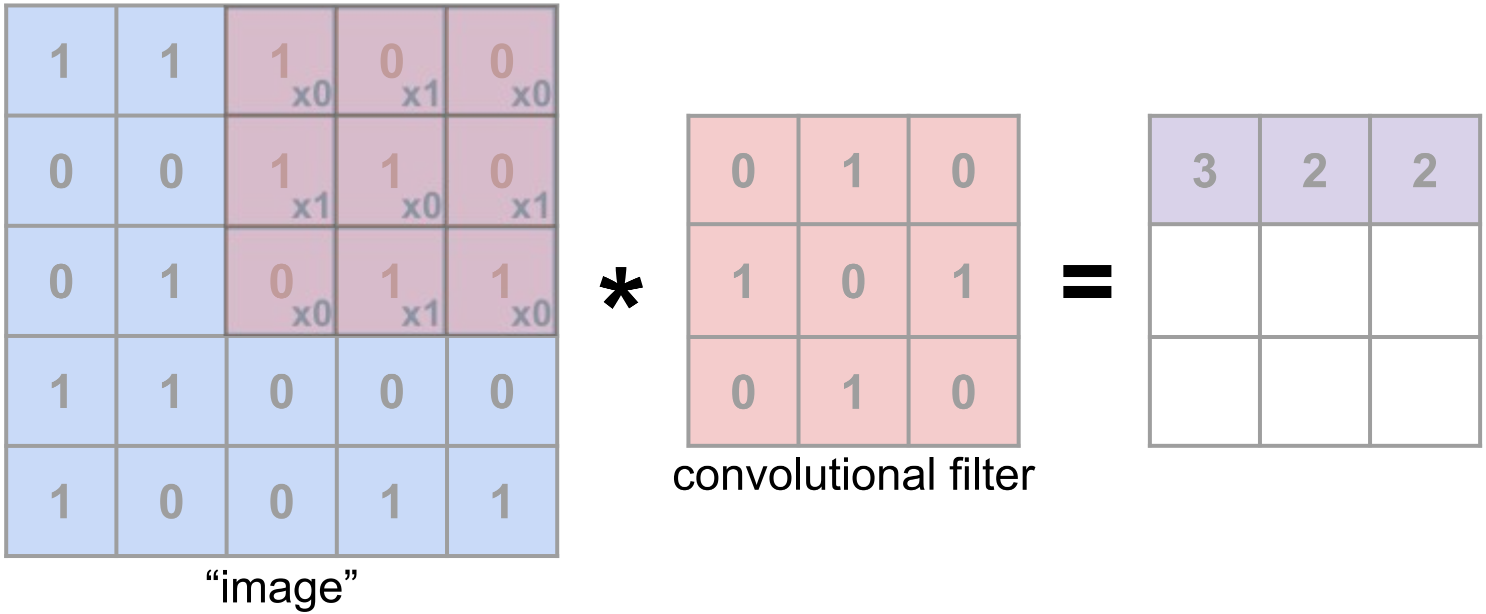


Cornell Bowers C·IS

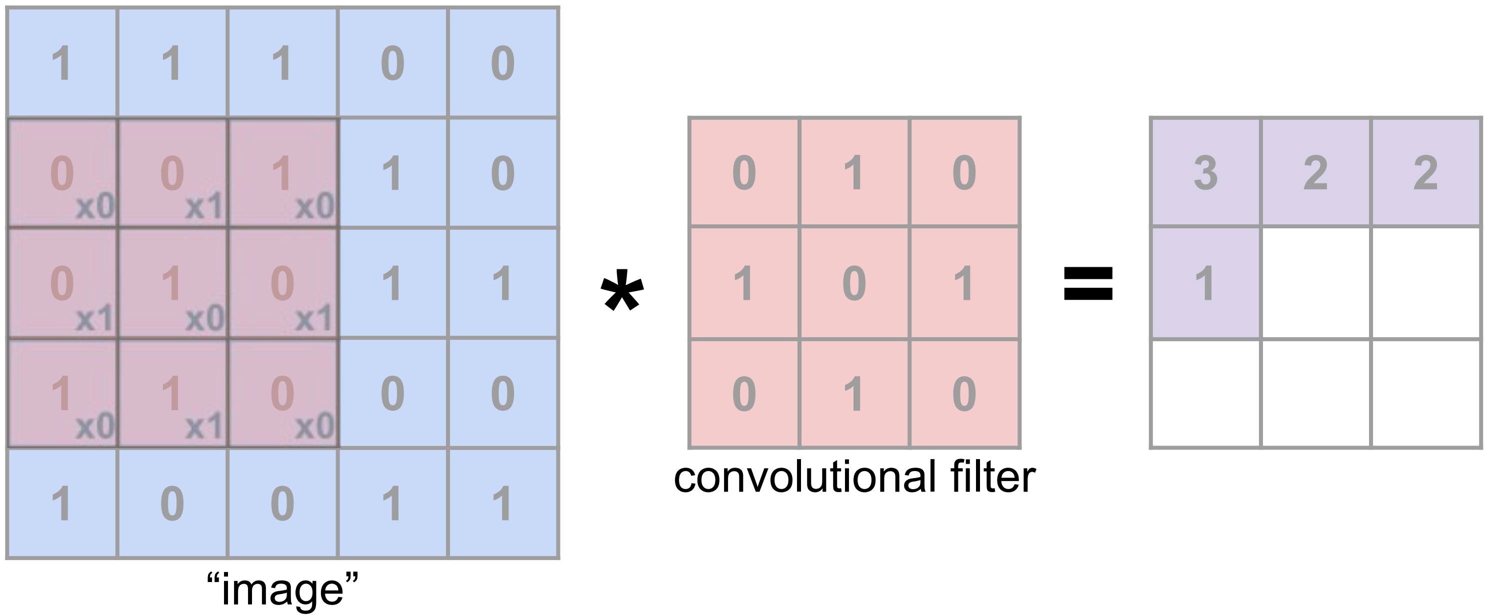
Convolutional Filters



Convolutional Filters

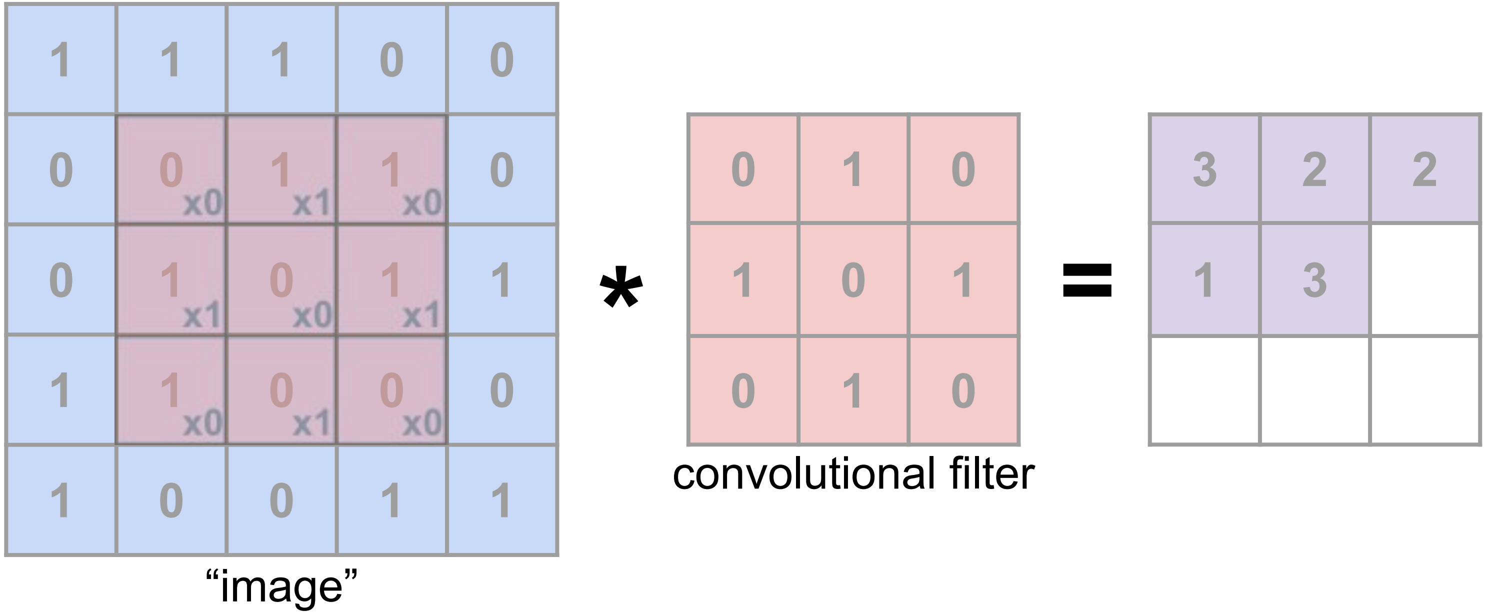


Convolutional Filters



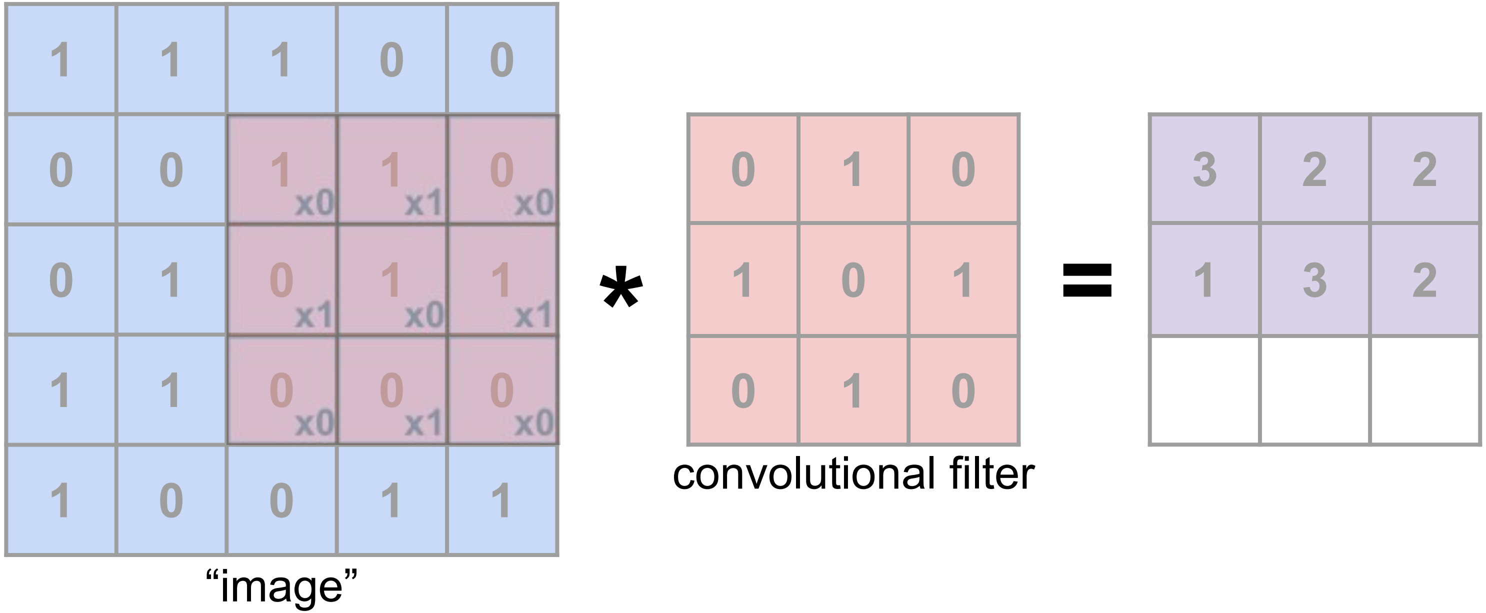
Cornell Bowers C·IS

Convolutional Filters

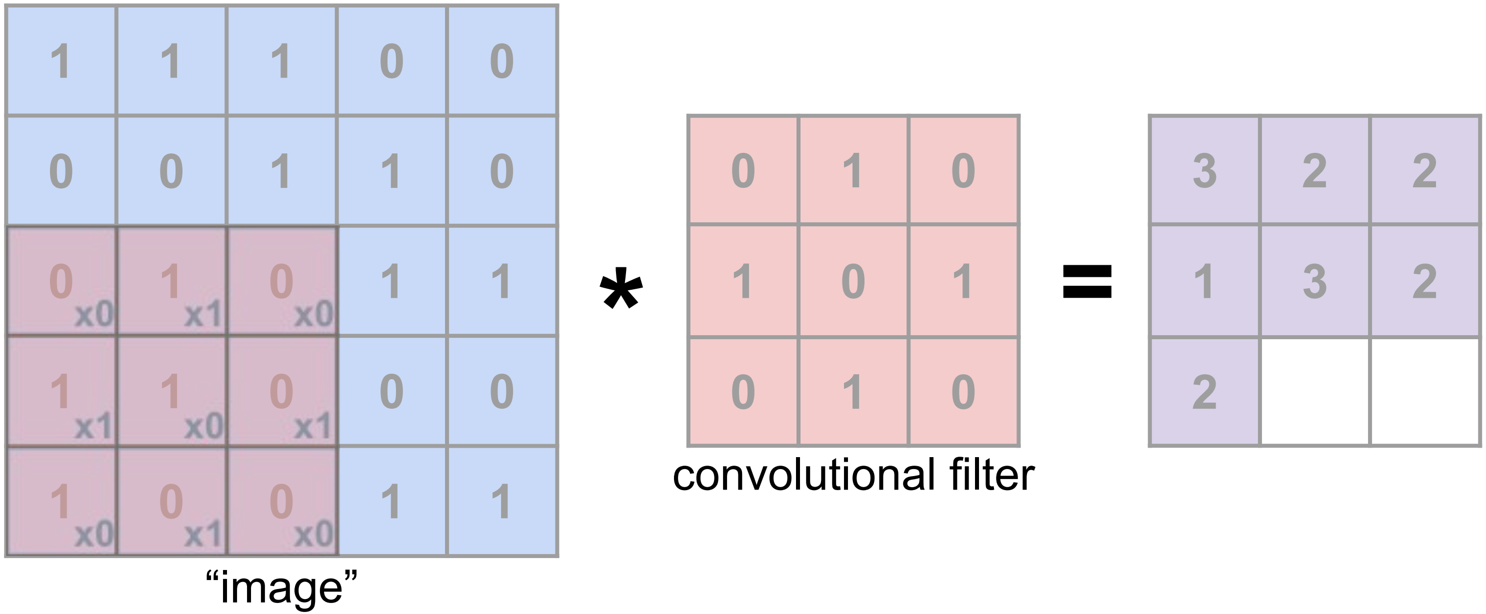


Cornell Bowers C·IS

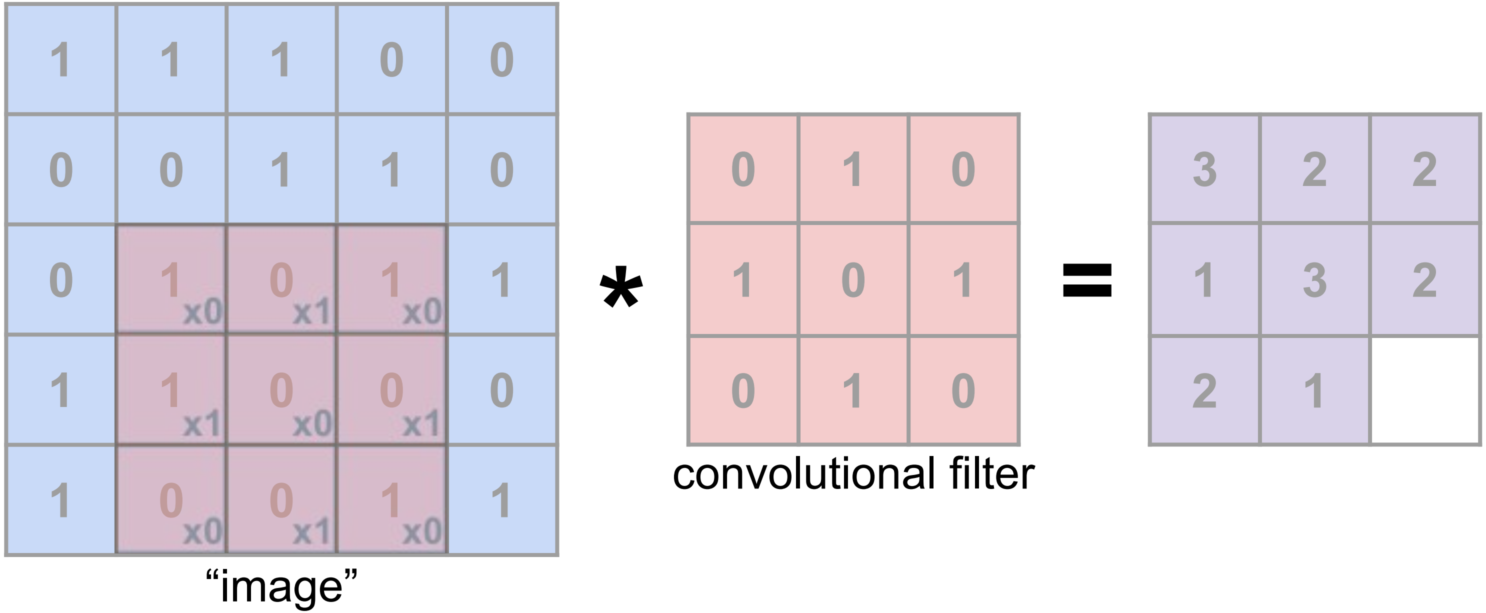
Convolutional Filters



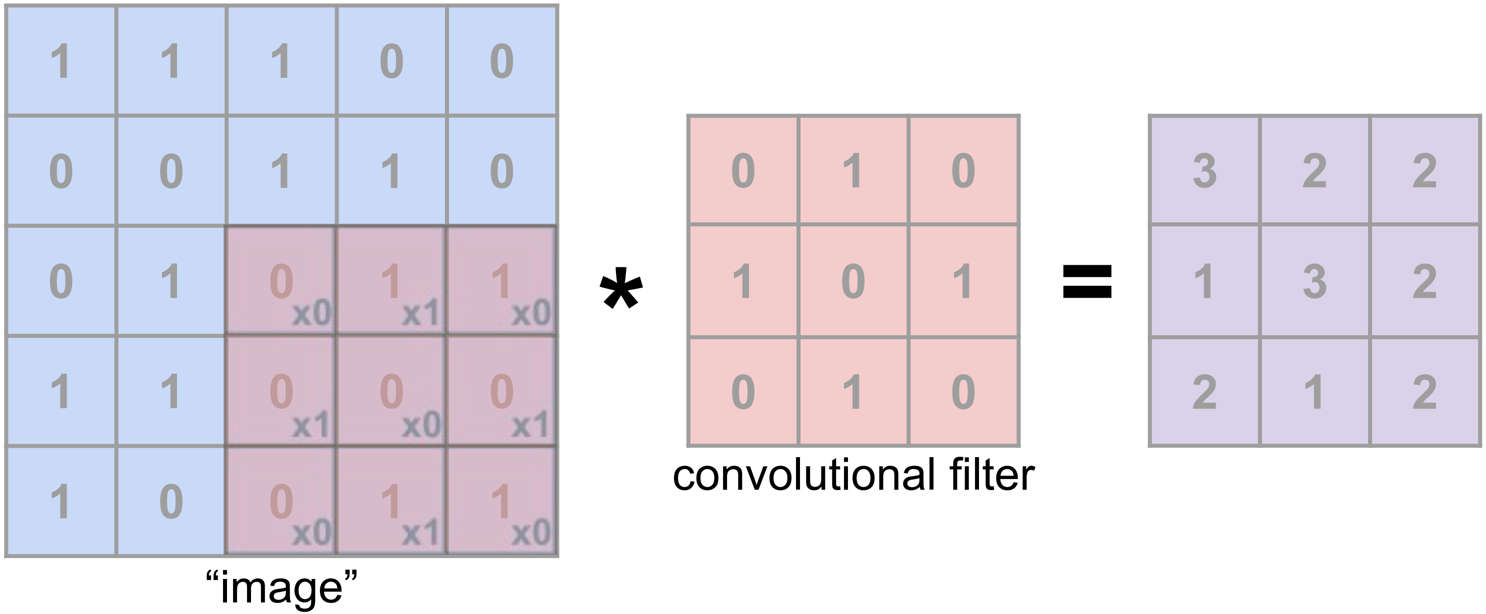
Convolutional Filters



Convolutional Filters

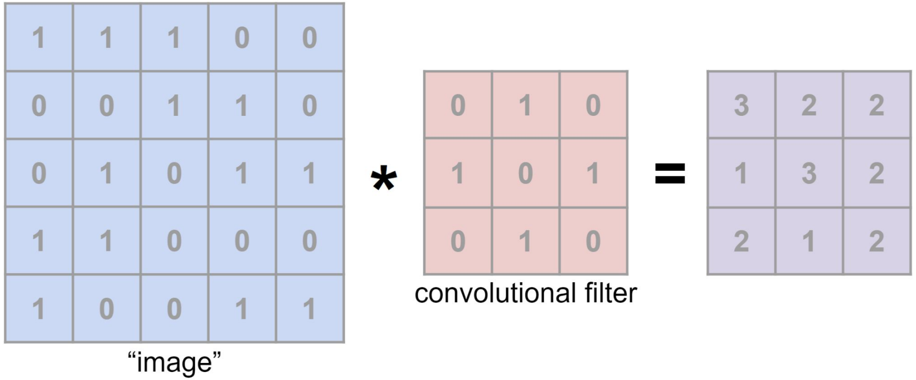


Convolutional Filters



Convolutional Filters

- ❖ Aggregates information from local window around pixel
- ❖ Translational invariance
- ❖ Reduce number of parameters needed to be learned



Discuss with your Neighbor!

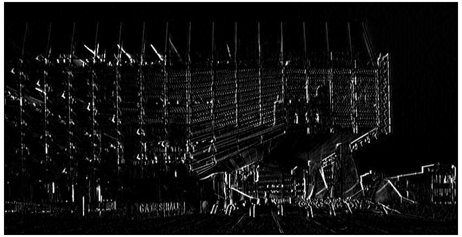
Match the following convolutional filters with the output they produce.

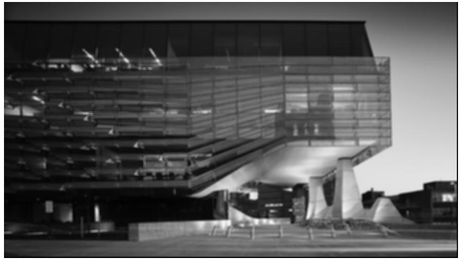
input image

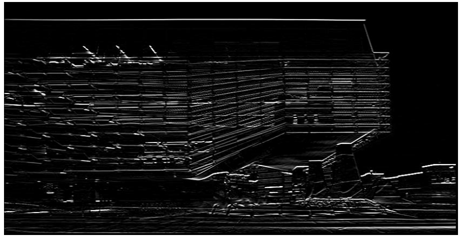
-1	-1	-1
0	0	0
1	1	1

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

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9







Discuss with your Neighbor!

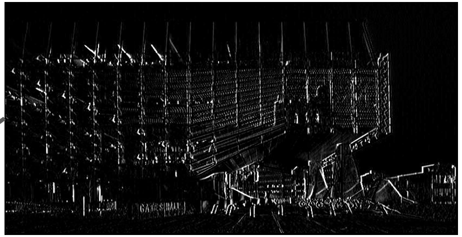
Match the following convolutional filters with the output they produce.

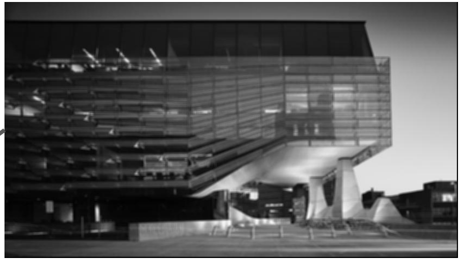
input image

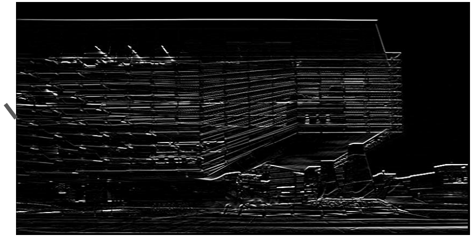
-1	-1	-1
0	0	0
1	1	1

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

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9







Cornell Bowers C·IS

Convolutional Filters

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 2 & 2 \\ 1 & 3 & 2 \\ 2 & 1 & 2 \end{bmatrix}$$

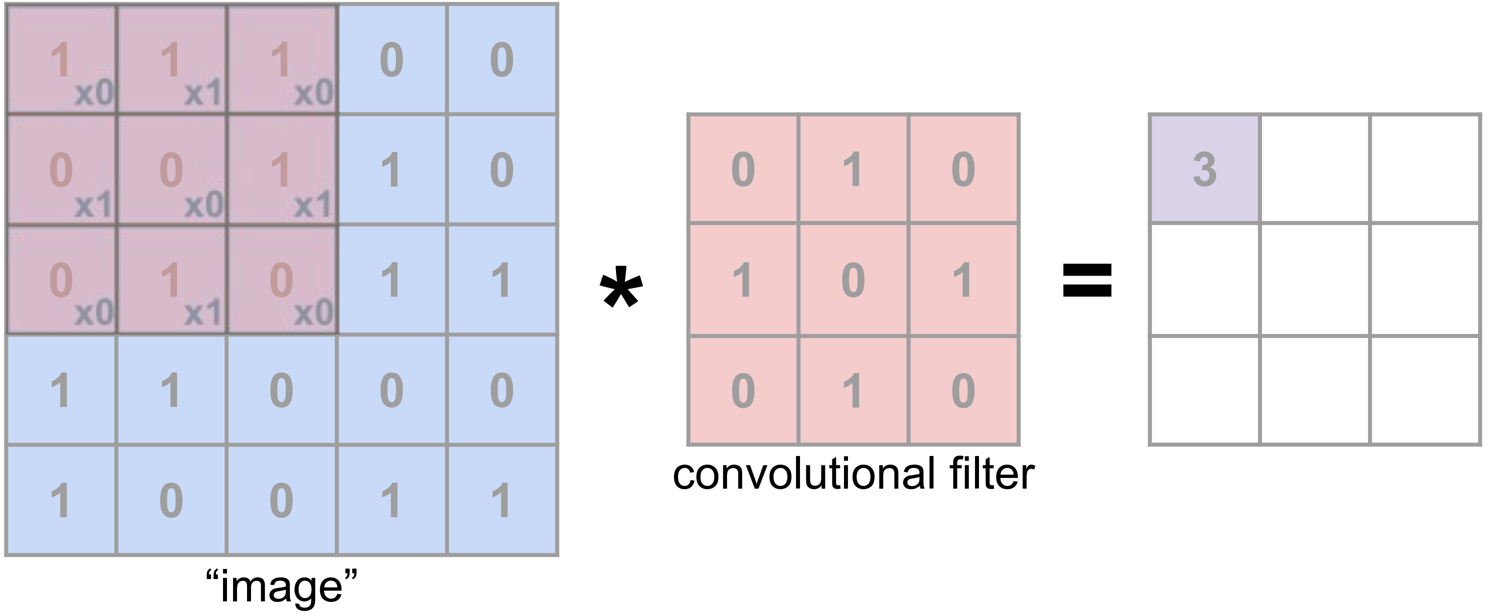
“image”

convolutional filter

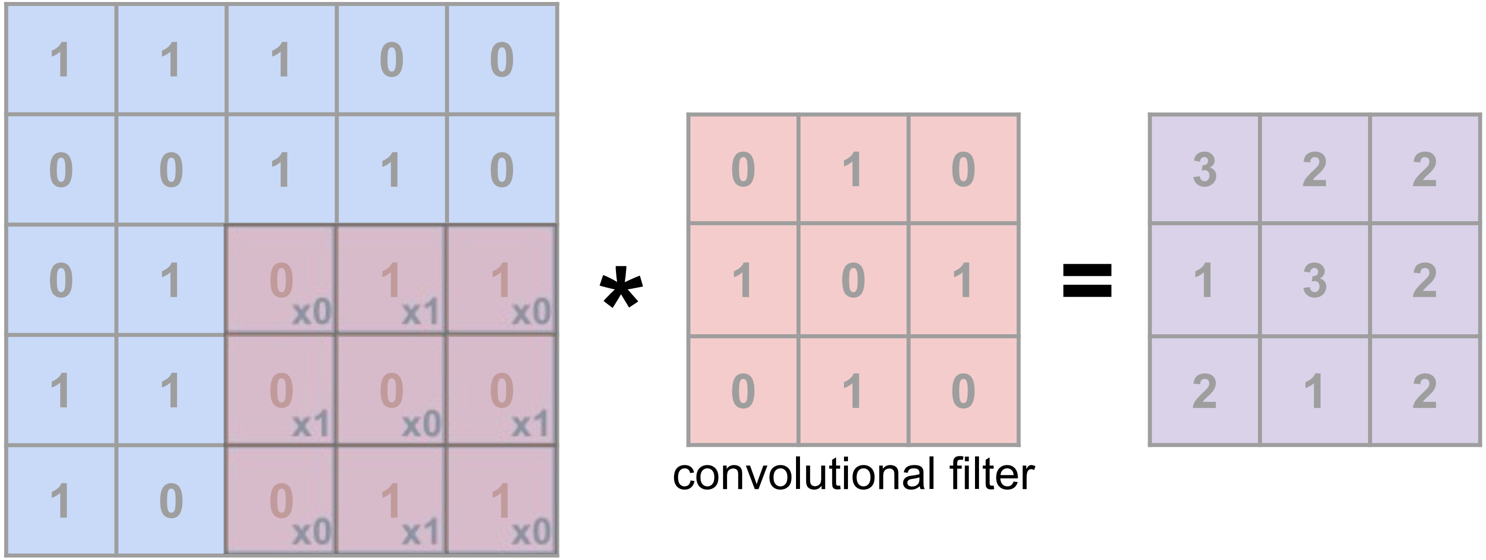
↓

can learn this!

Convolutional Filters



Convolutional Filters

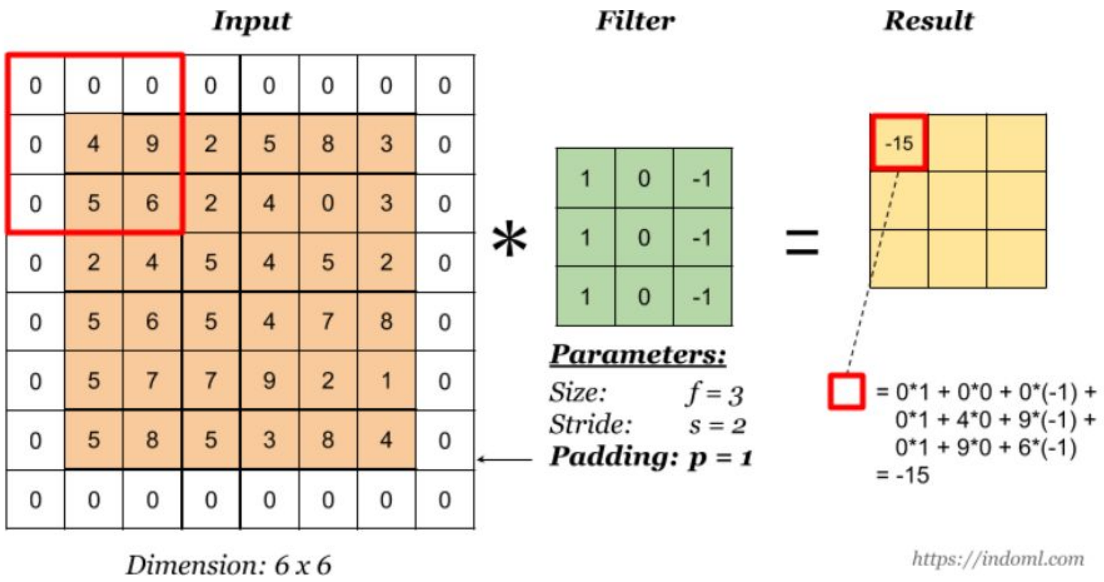


“image”

Cornell Bowers C·IS

CNNs - Padding

- Padding adds layers of zeros (or other number) around image border
- Prevents image shrinking and loss of information from image boundary



Cornell Bowers C·IS

CNNs - Padding

mode : *{'full', 'valid', 'same'}, optional*

'full':

By default, mode is 'full'. This returns the convolution at each point of overlap, with an output shape of (N+M-1,). At the end-points of the convolution, the signals do not overlap completely, and boundary effects may be seen.

'same':

Mode 'same' returns output of length `max(M, N)`. Boundary effects are still visible.

'valid':

Mode 'valid' returns output of length `max(M, N) - min(M, N) + 1`. The convolution product is only given for points where the signals overlap completely. Values outside the signal boundary have no effect.

Dimension: 6 x 6

https://indoml.com

CNNs - Padding

- **mode : *{'full', 'valid', 'same'}, optional***

  'full':

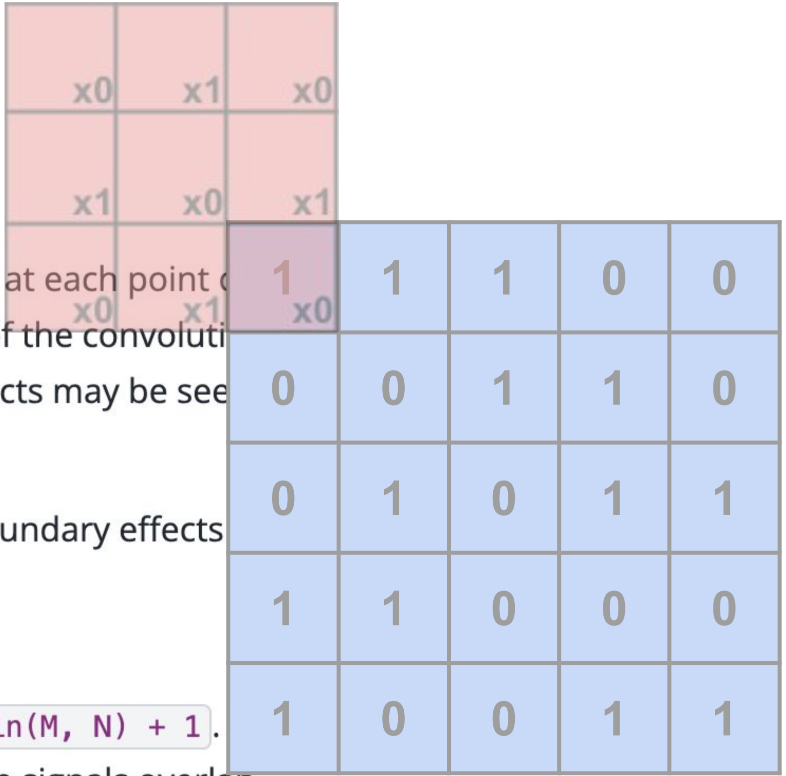
  By default, mode is 'full'. This returns the convolution at each point
  with an output shape of (N+M-1,). At the end-points of the convoluti
  signals do not overlap completely, and boundary effects may be see

  'same':

  Mode 'same' returns output of length `max(M, N)`. Boundary effects
  visible.

  'valid':

  Mode 'valid' returns output of length `max(M, N) - min(M, N) + 1`.
  convolution product is only given for points where the signals overlap
  completely. Values outside the signal boundary have no effect.

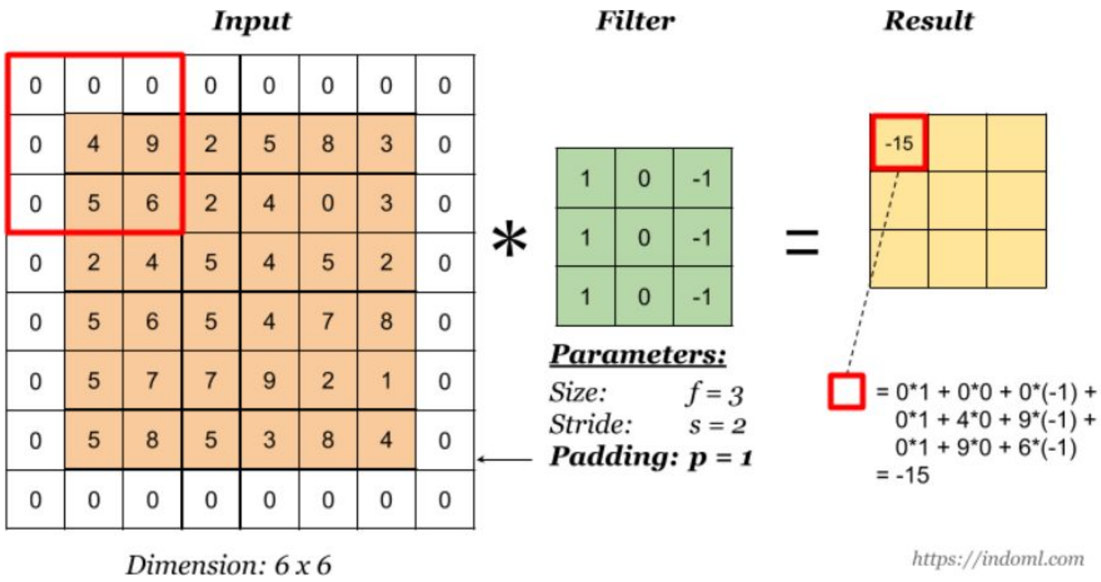


Dimension: 6 x 6

https://indoml.com

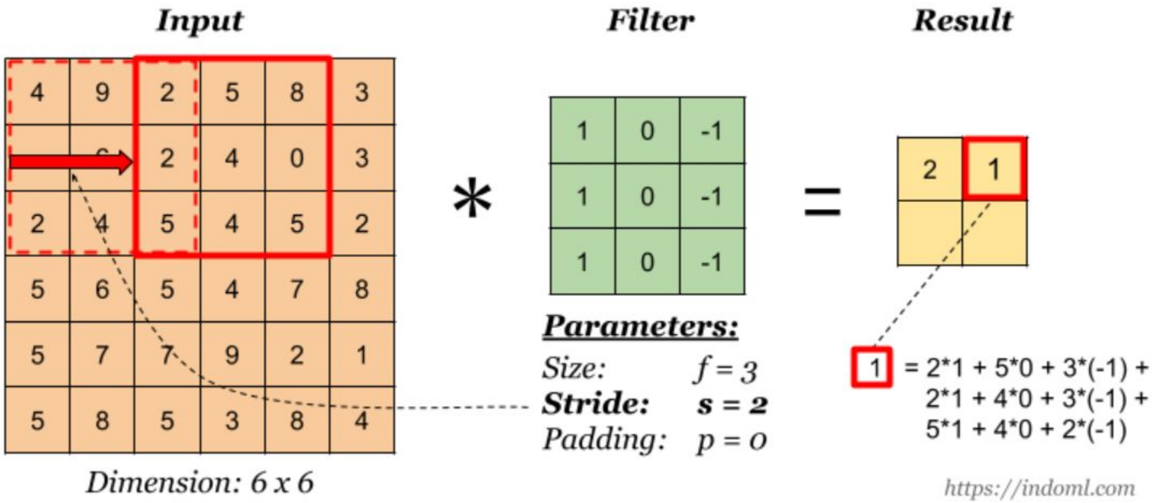
CNNs - Padding

- `padding` controls the amount of padding applied to the input. It can be either a string {'valid', 'same'} or an int / a tuple of ints giving the amount of implicit padding applied on both sides.



CNNs - Stride

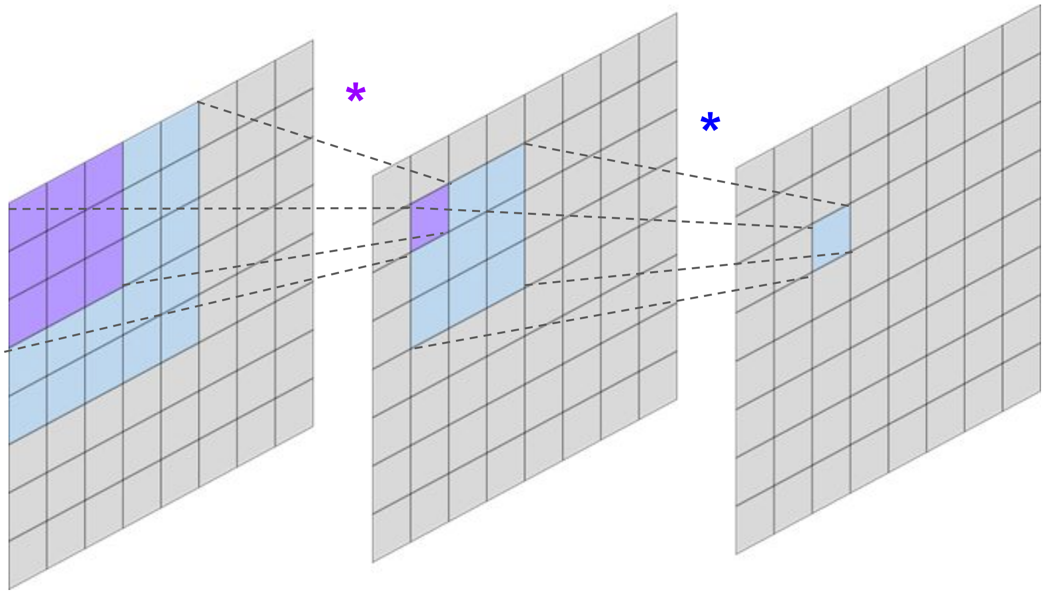
- Stride controls how many units the filter / the receptive field shift at a time
- The size of the output image shrinks more as the stride becomes larger
- The receptive fields overlap less as the stride becomes larger



Filter with stride (s) = 2

Cornell Bowers C·IS

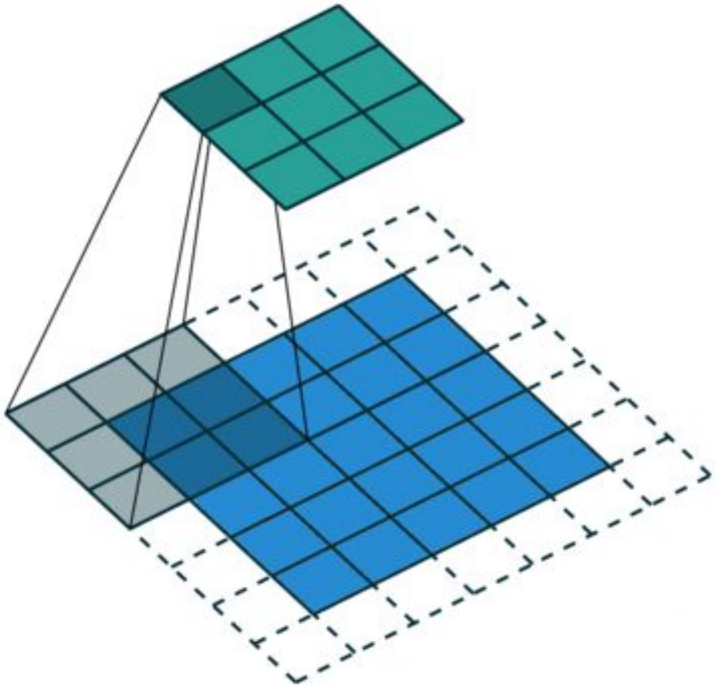
Stacking Convolutions

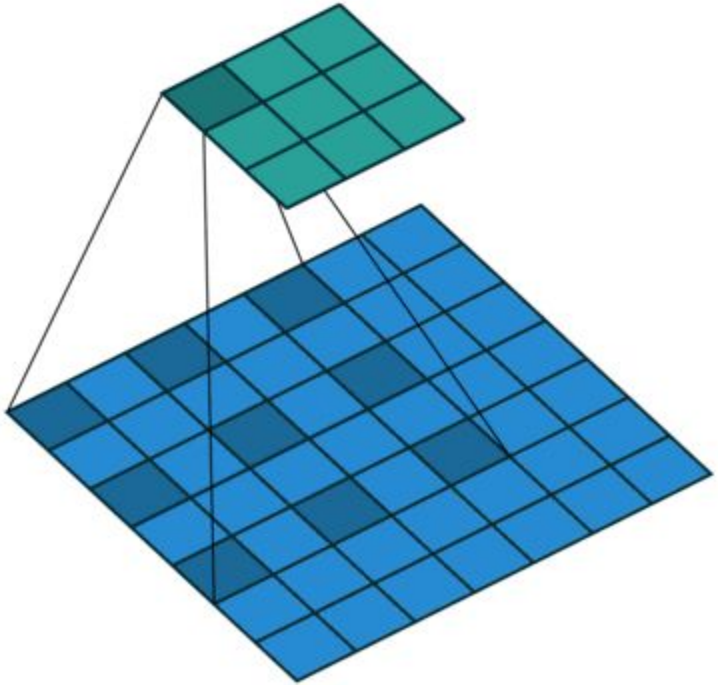


- Size of receptive field increases with each layer
- Capture more complex features

Cornell Bowers C·IS

Dilated Convolutions





https://github.com/vdumoulin/conv_arithmetic

Cornell Bowers C·IS

Dilated convolution or Atrous convolution?

SEMANTIC IMAGE SEGMENTATION WITH DEEP CONVOLUTIONAL NETS AND FULLY CONNECTED CRFS

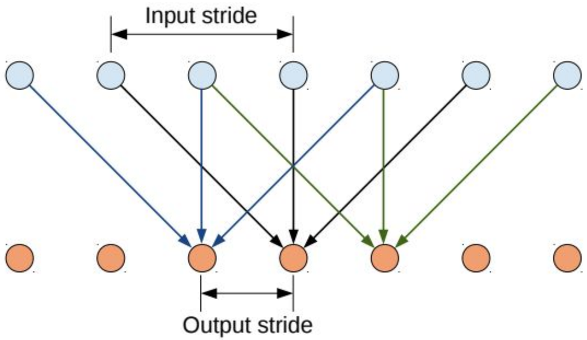
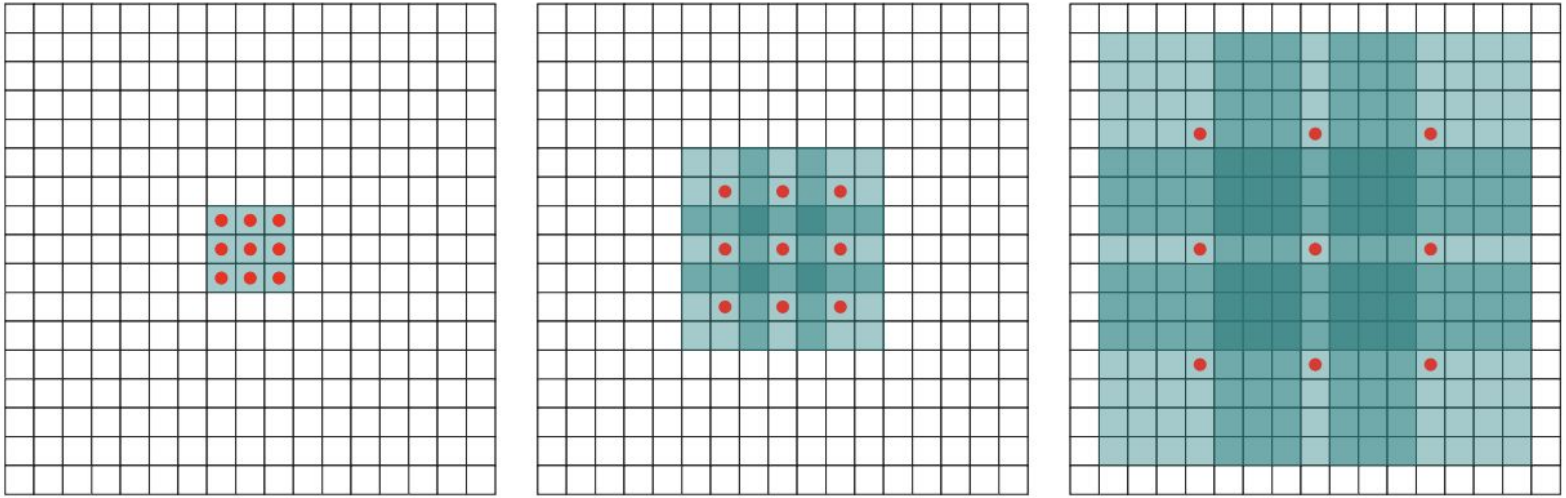


Figure 1: Illustration of the hole algorithm in 1-D, when *kernel_size* = *3*, *input_stride* = *2*, and *output_stride* = *1*.

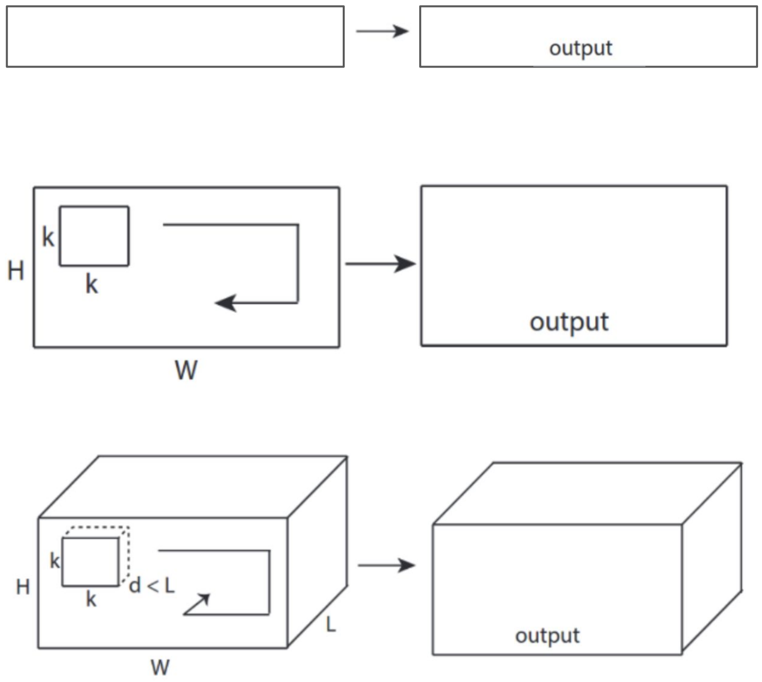
Dilated convolution or Atrous convolution?

Multi-Scale Context Aggregation by Dilated Convolutions



(a) (b) (c)

1D and 3D Convolutions



https://wandb.ai/ayush-thakur/dl-question-bank/reports/Intuitive-understanding-of-1D-2D-and-3D-convolutions-in-convolutional-neural-networks---VmlldzoxOTk2MDA

Cornell Bowers C·IS

Convolutional Filters

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

"image"

*

0	1	0
1	0	1
0	1	0

convolutional filter

↓

can learn this!

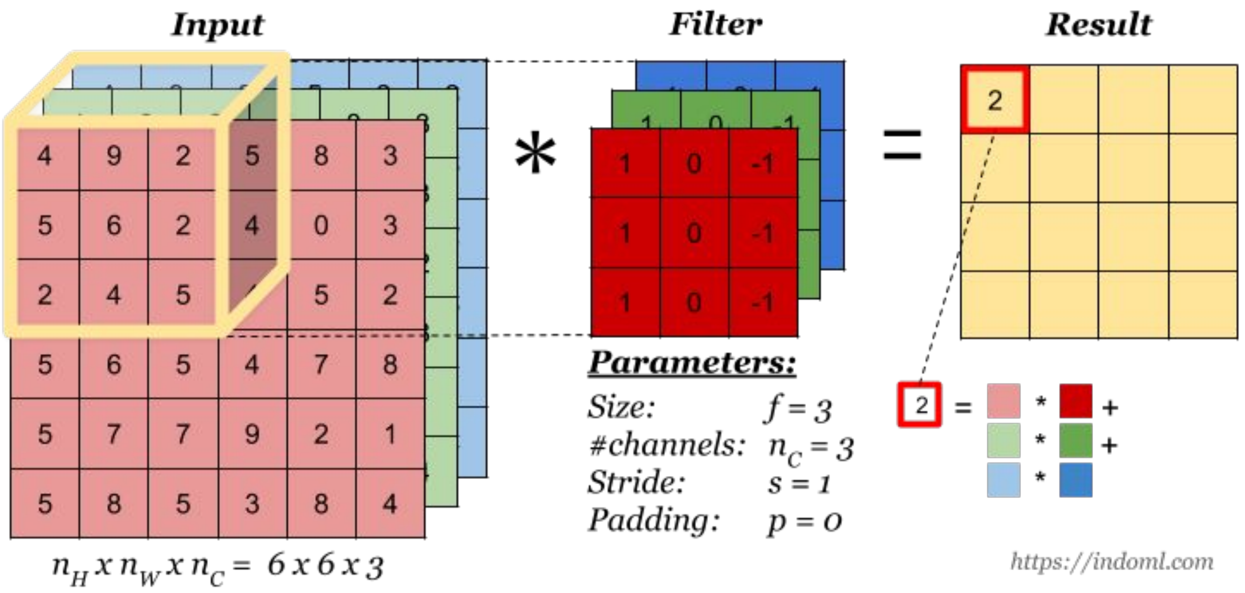
=

3	2	2
1	3	2
2	1	2

Cornell Bowers C·IS

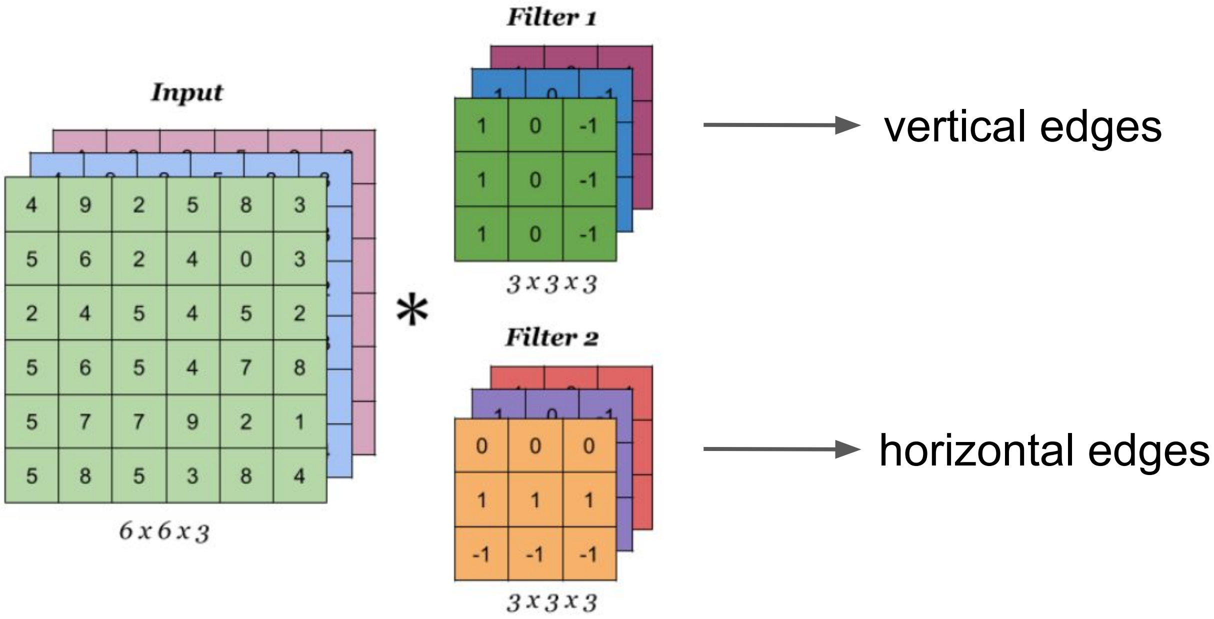
Multiple Input Channels: Convolution Over Volumes

What if our input image has more than one channel?



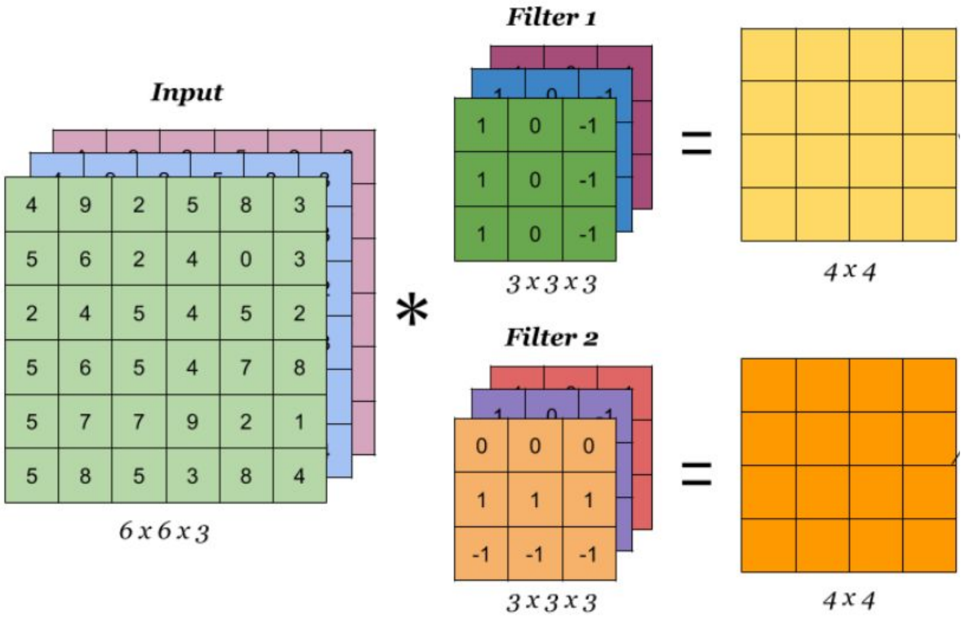
Cornell Bowers C·IS

Multiple Output Channels: Multiple Filters



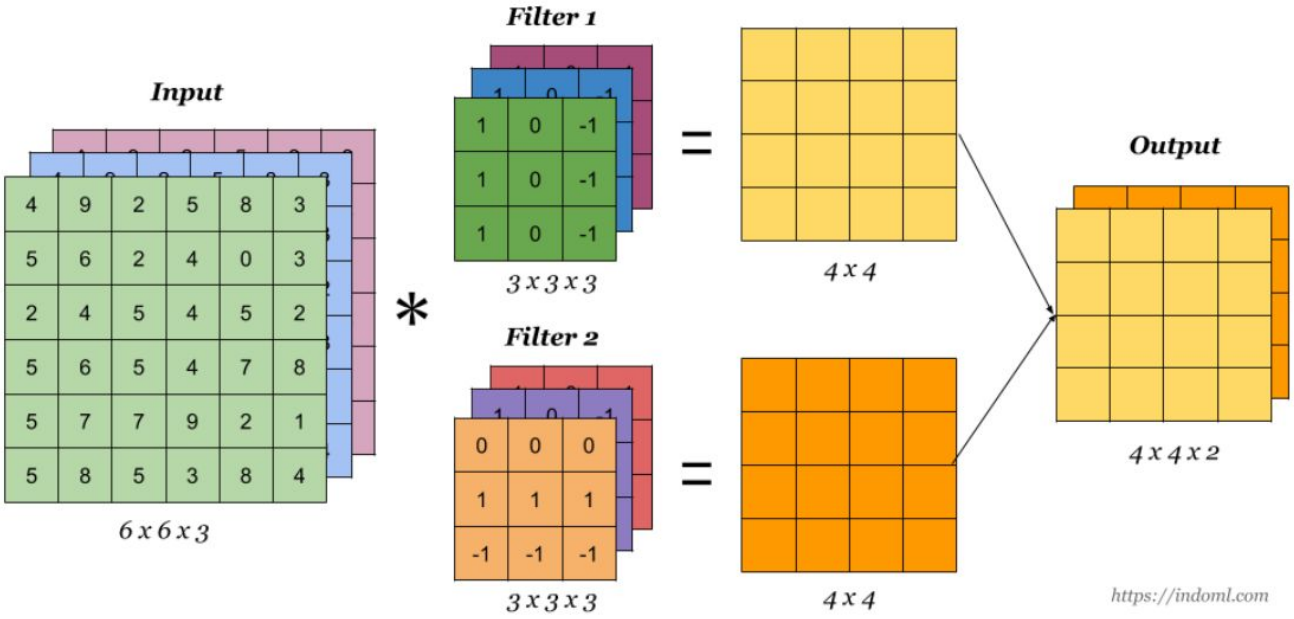
Cornell Bowers C·IS

Multiple Output Channels: Multiple Filters



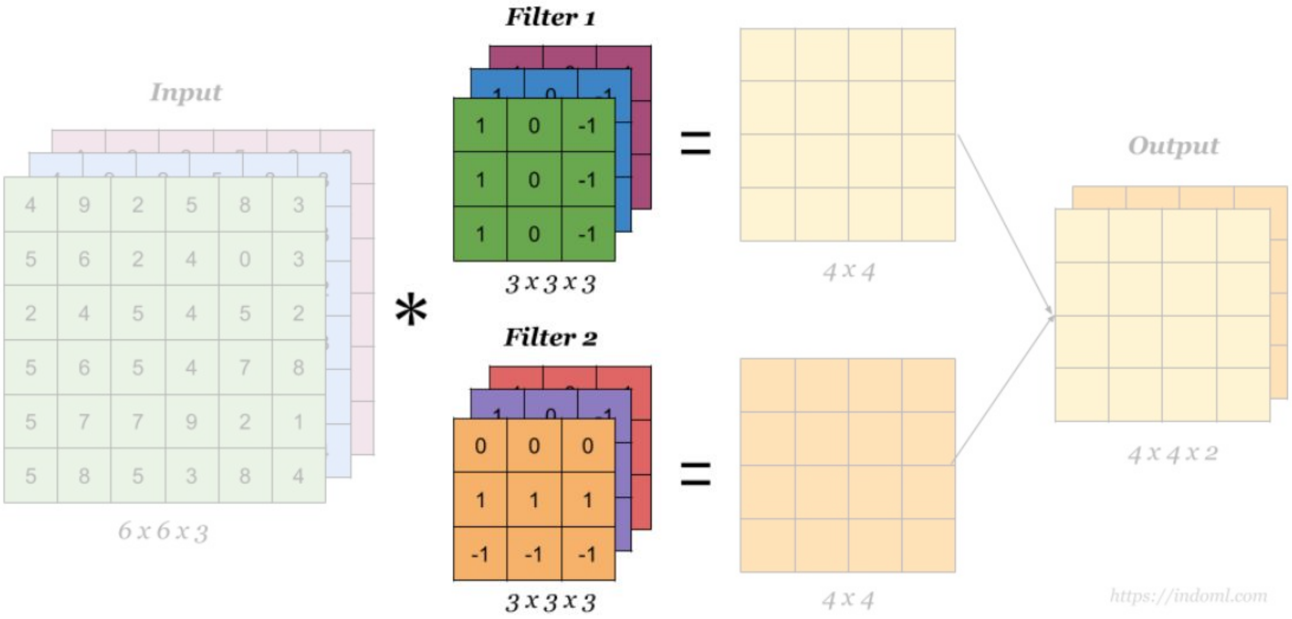
Cornell Bowers C·IS

Multiple Output Channels: Multiple Filters



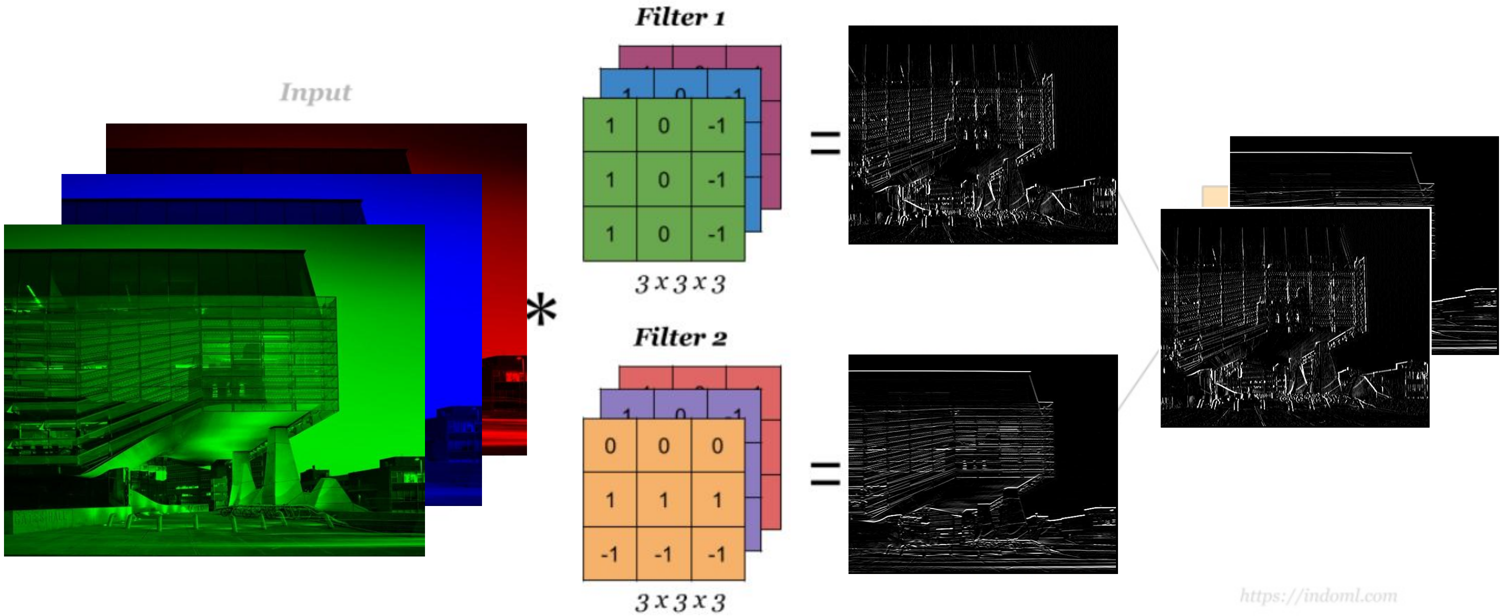
Cornell Bowers C·IS

Multiple Output Channels: Multiple Filters

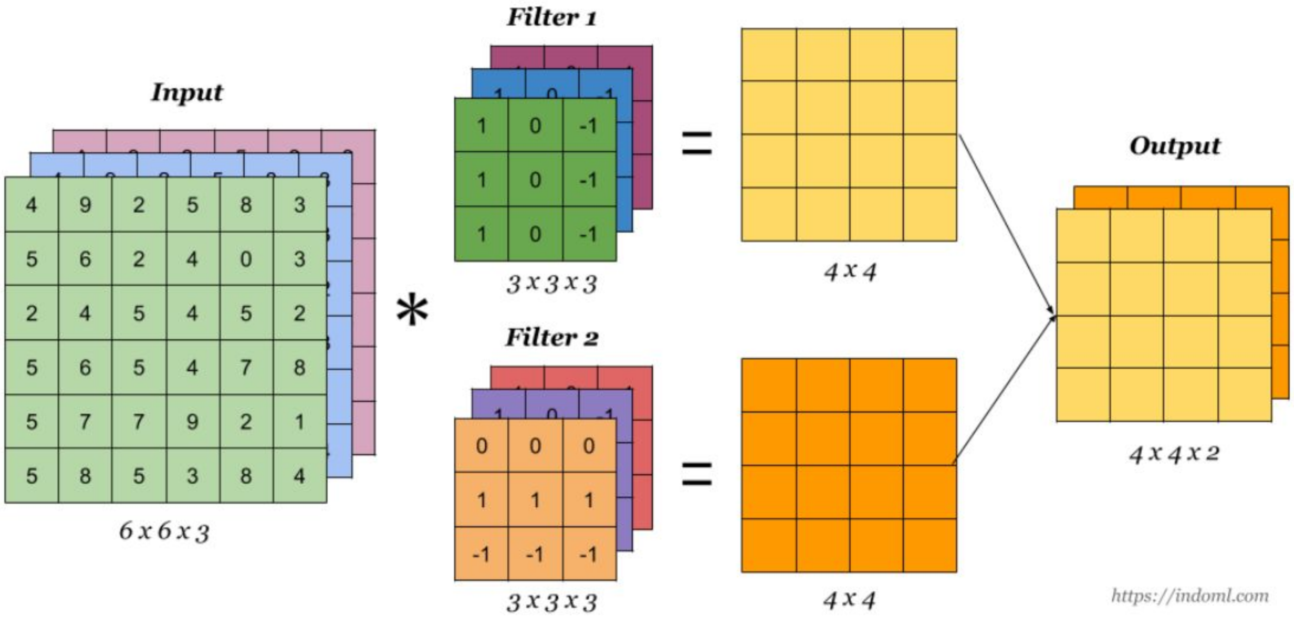


Cornell Bowers C·IS

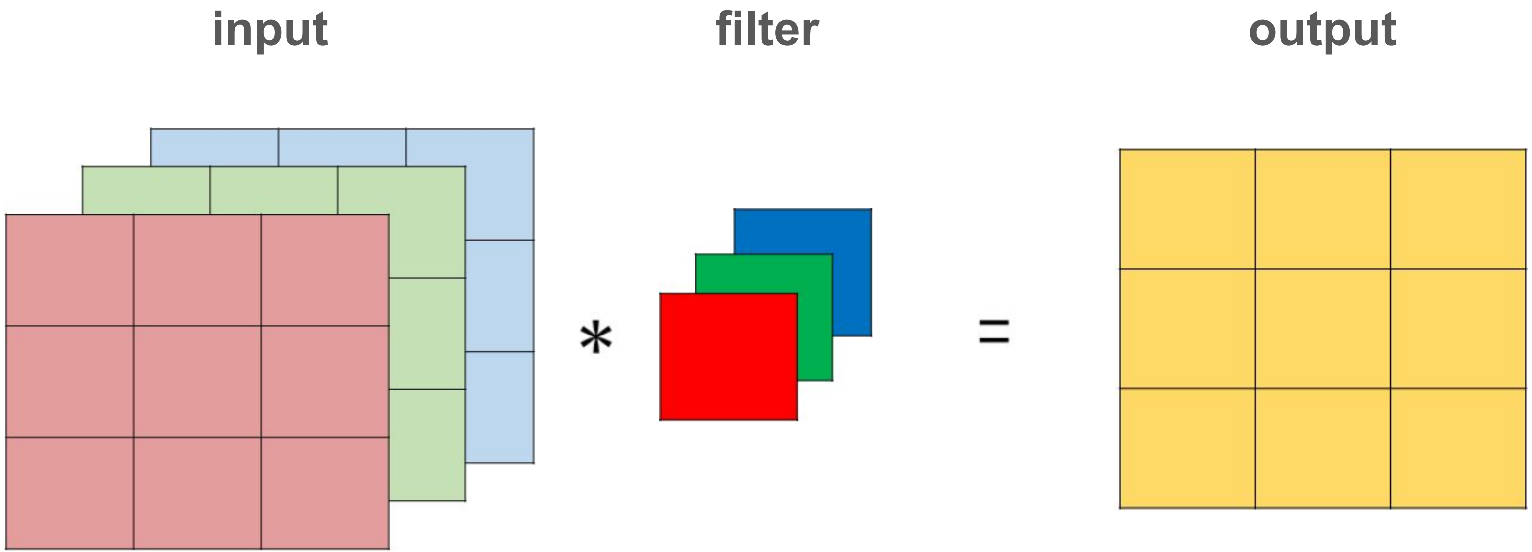
Multiple Output Channels: Multiple Filters



Multiple Output Channels: Multiple Filters

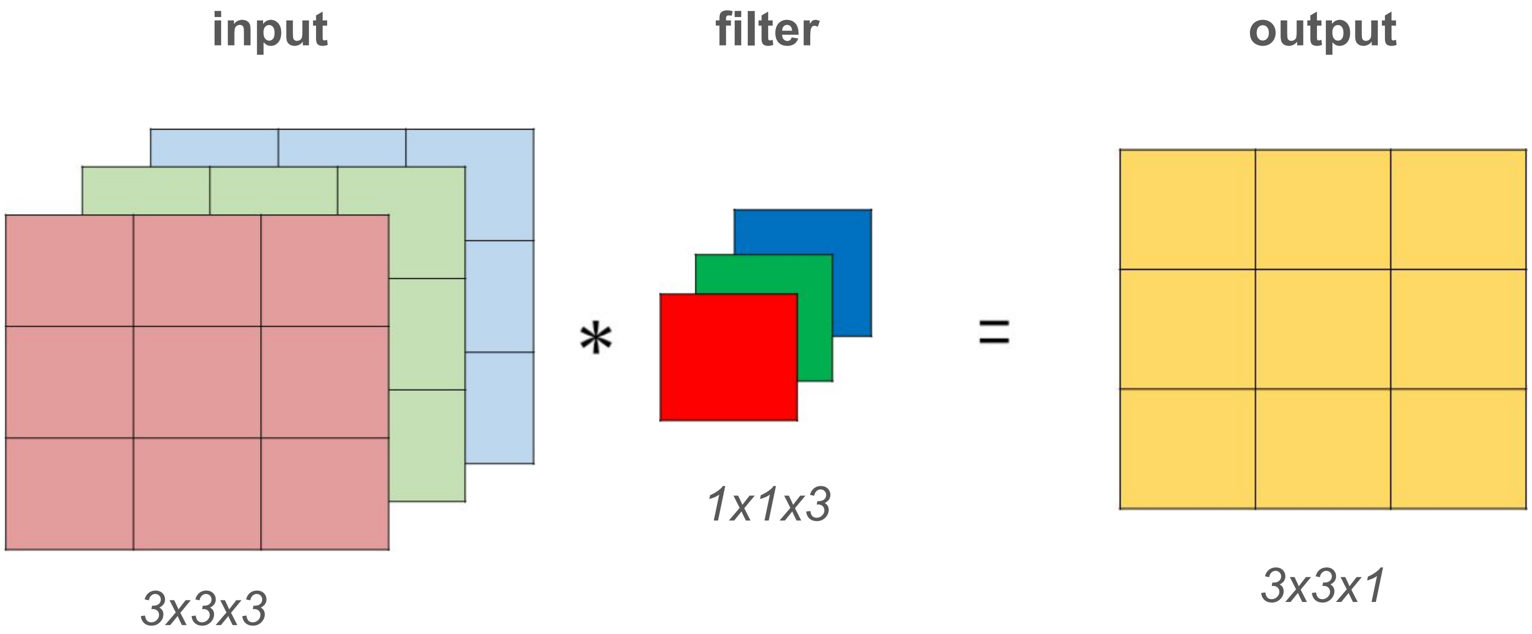


Slight Detour: 1x1 convolutions



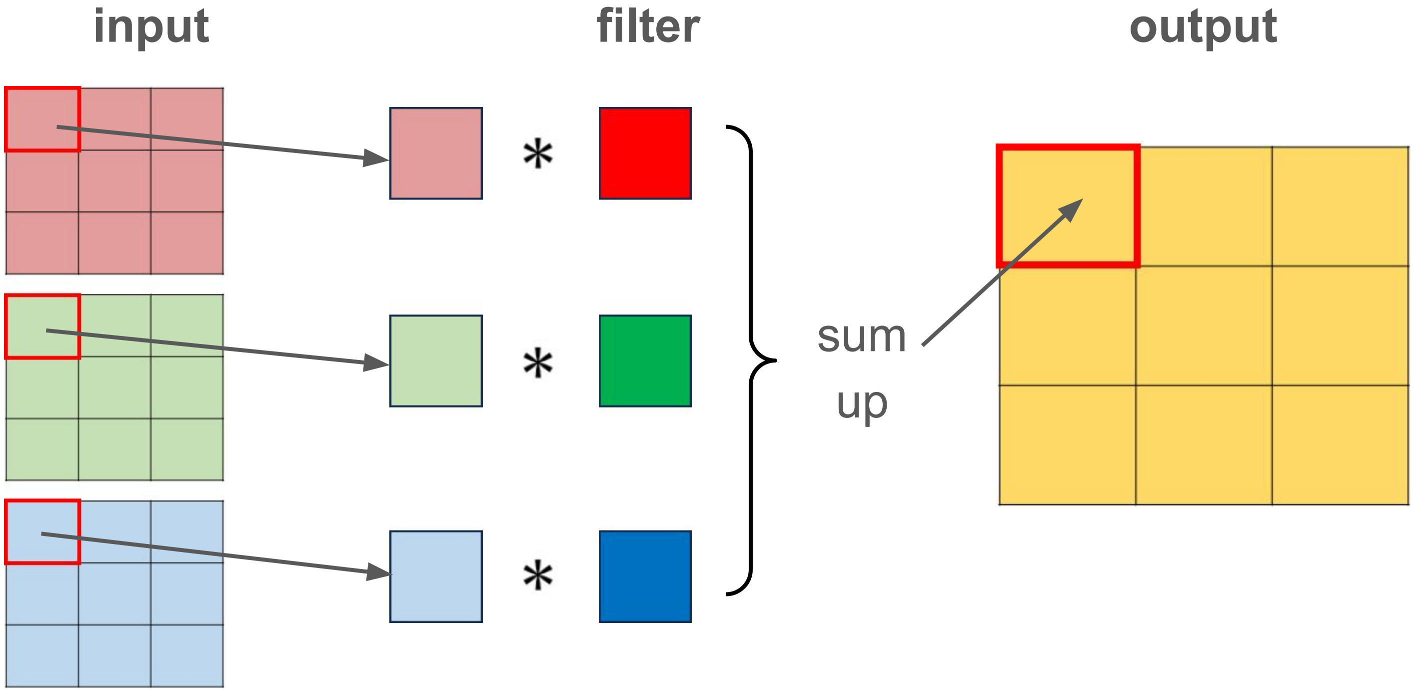
Cornell Bowers C·IS

Slight Detour: 1x1 convolutions



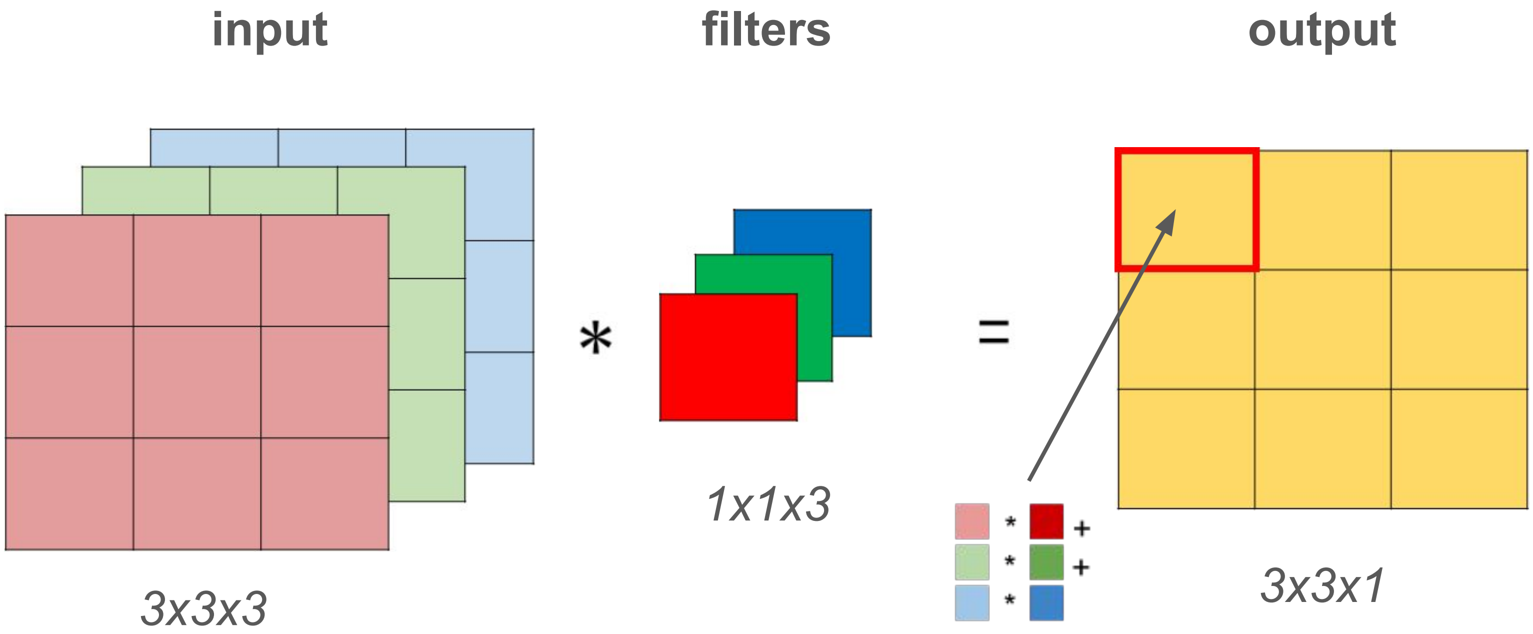
Cornell Bowers C·IS

Slight Detour: 1x1 convolutions



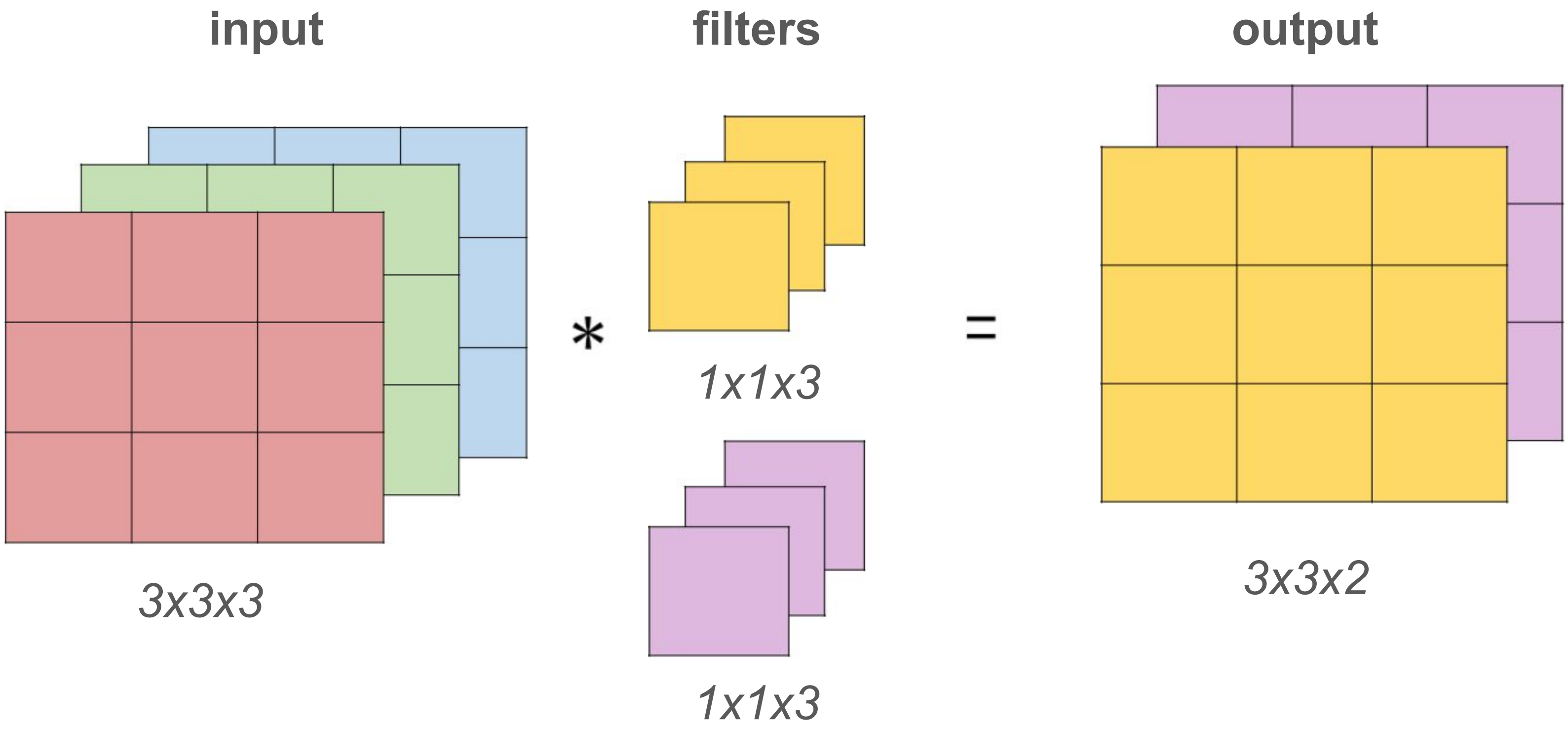
Cornell Bowers CIS

Slight Detour: 1x1 convolutions



Cornell Bowers C·IS

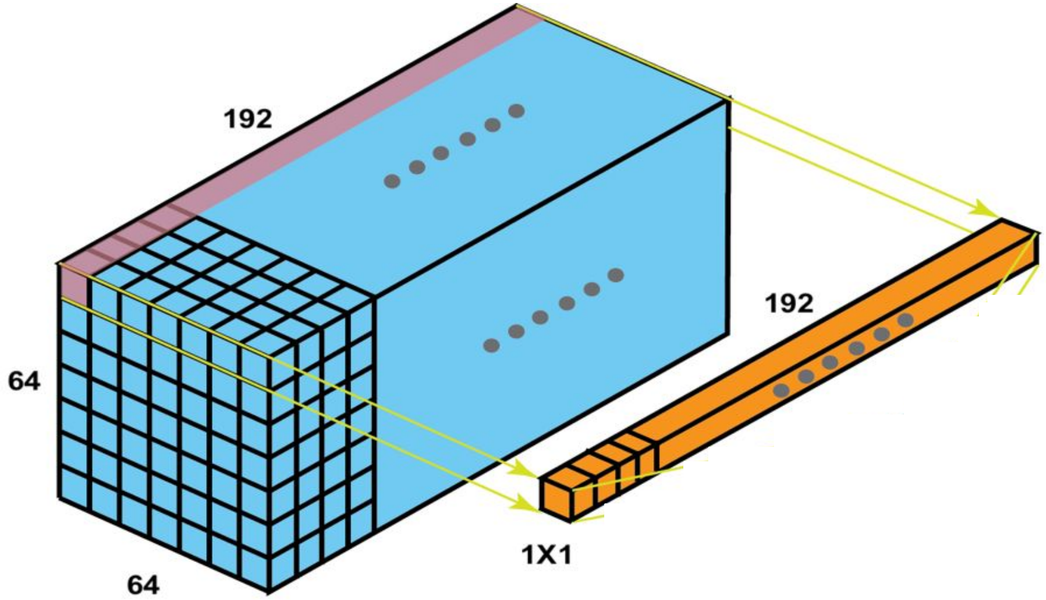
Slight Detour: 1x1 convolutions



Cornell Bowers C·IS

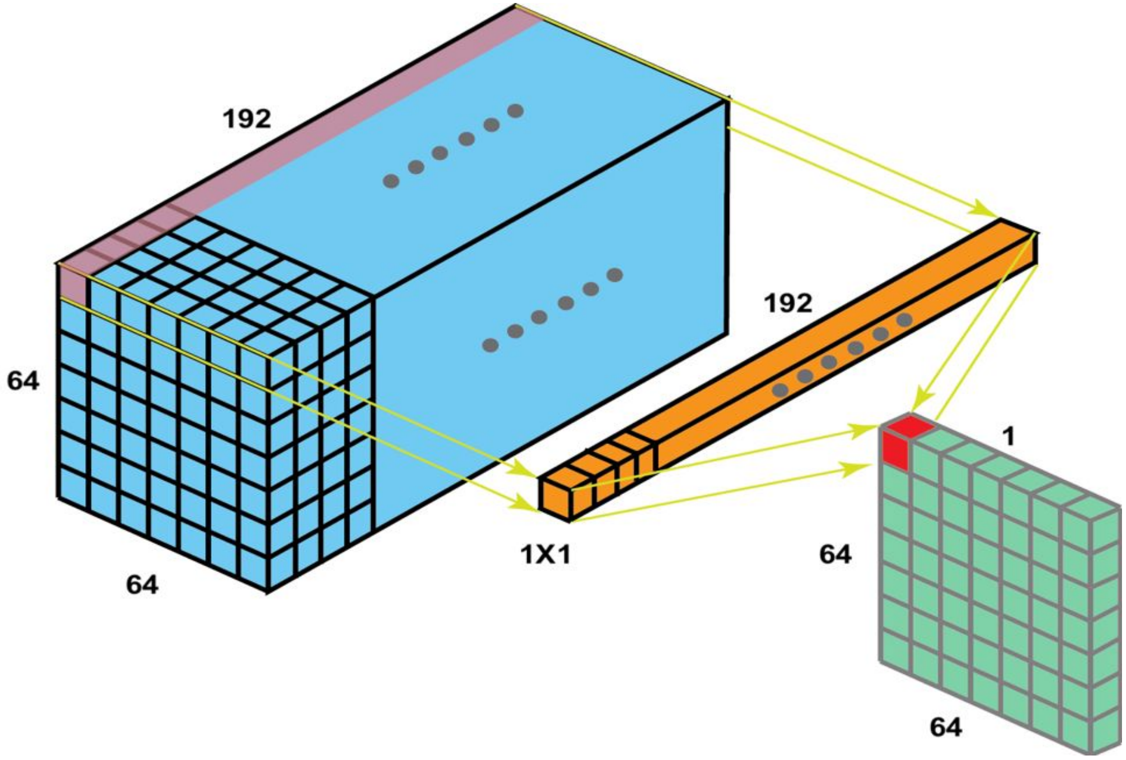
Discuss: 1x1 Convolutions

What is the result of convolving a 64x64x192 dimensional cube with a 1x1 filter?



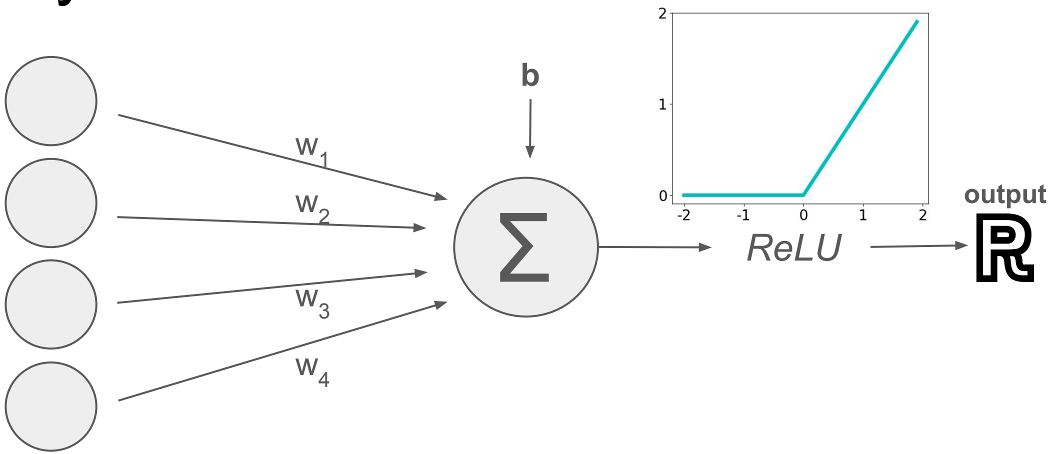
Cornell Bowers C·IS

1x1 Convolutions

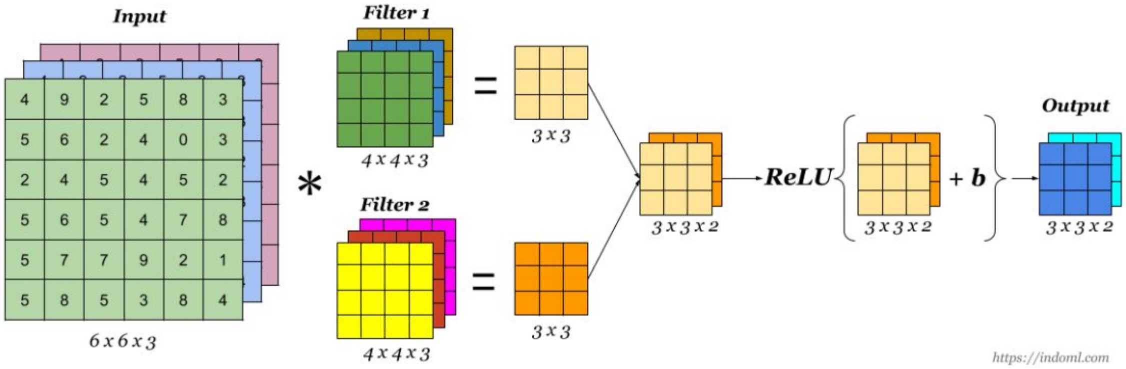


Convolution Layer

MLP Layer



Convolution Layer



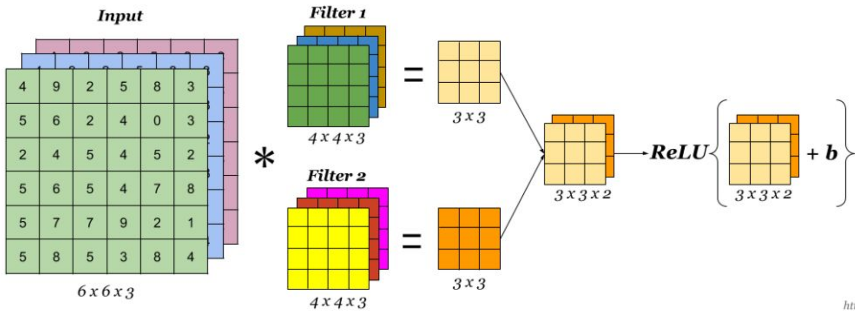
Cornell Bowers C·IS

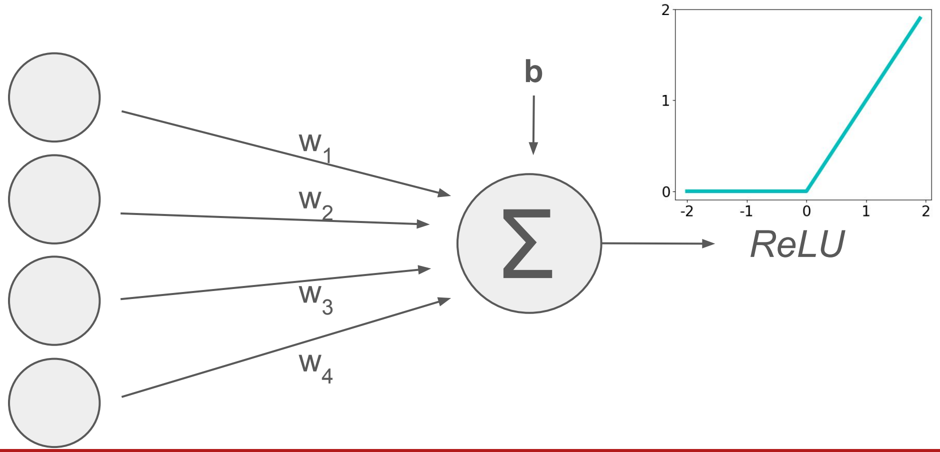
CNN/MLP Equivalence

Differences in a convolution layer:

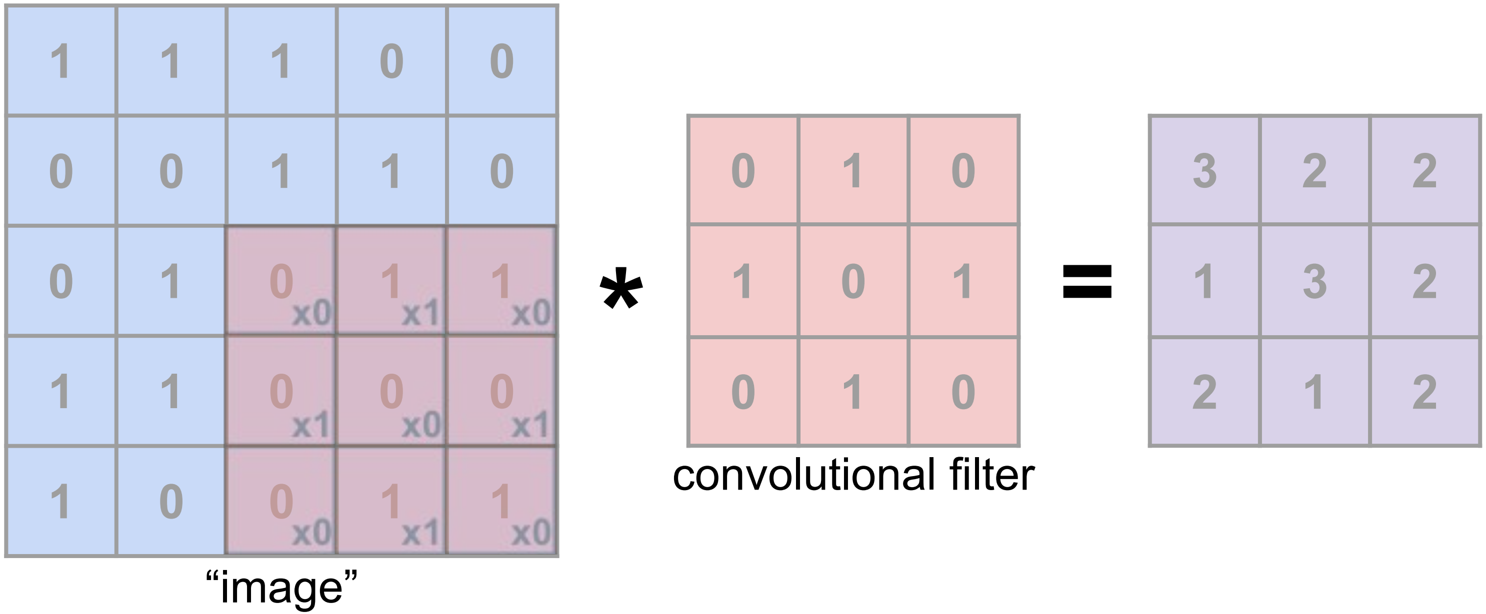
- neurons are connected to a local region
- Weights are shared across multiple parameters

CONV layers can be converted to Fully connected layers and vice versa!



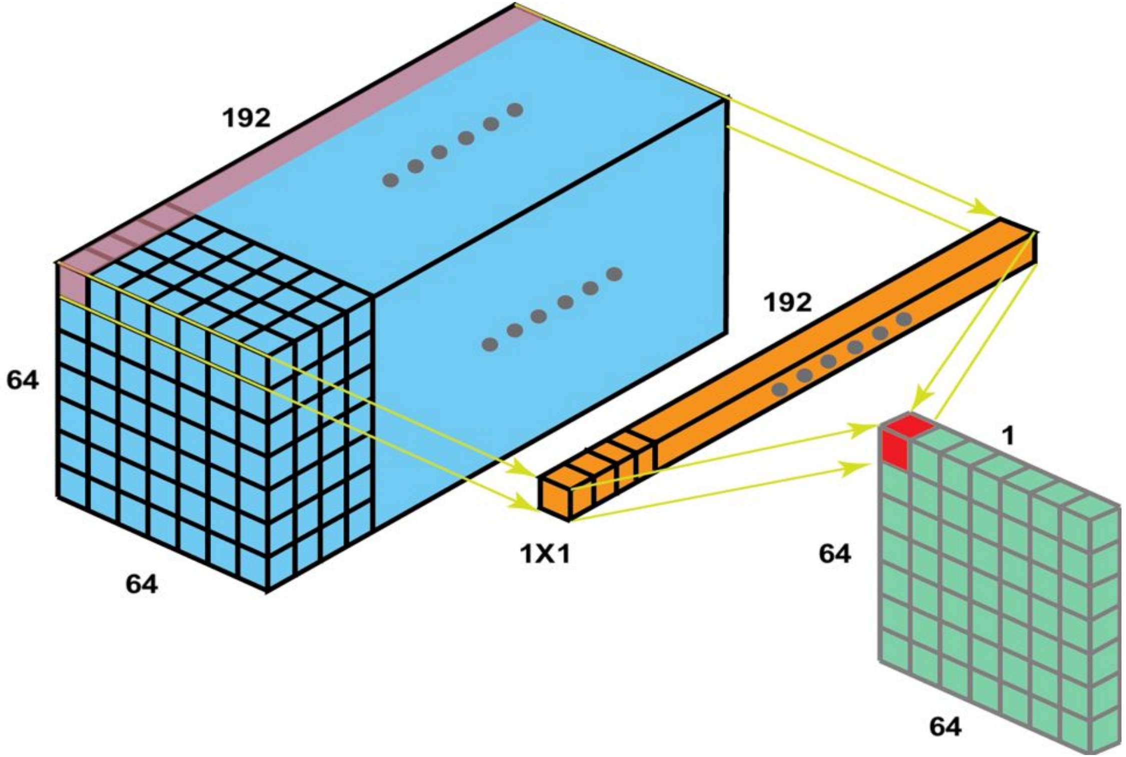


Convolutional Filters



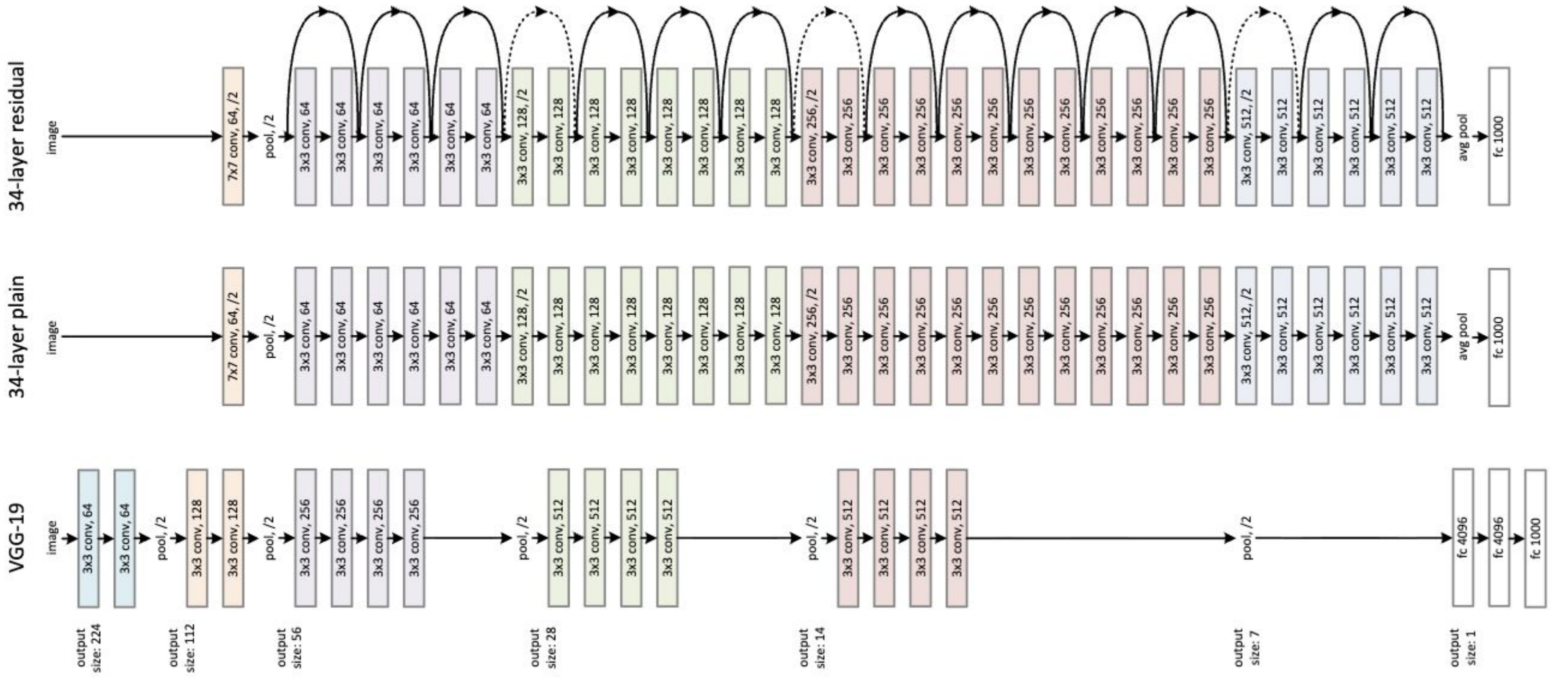
Discuss: Trade-offs between CNNs and MLPs

How would this image change if you used an MLP instead of a 1x1 convolution filter to produce a (64x64x1) feature map? Hint: think about parameter counts and feature interactions.



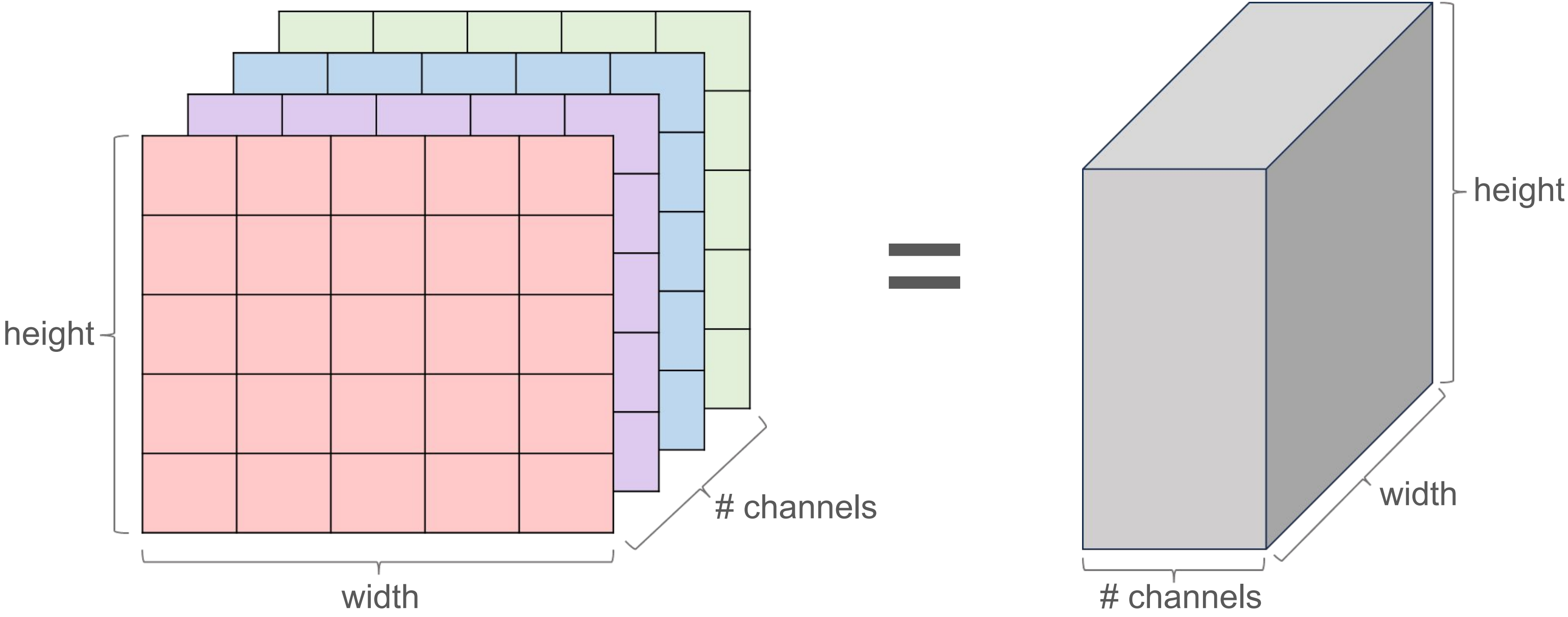
Cornell Bowers C·IS

Our goal today



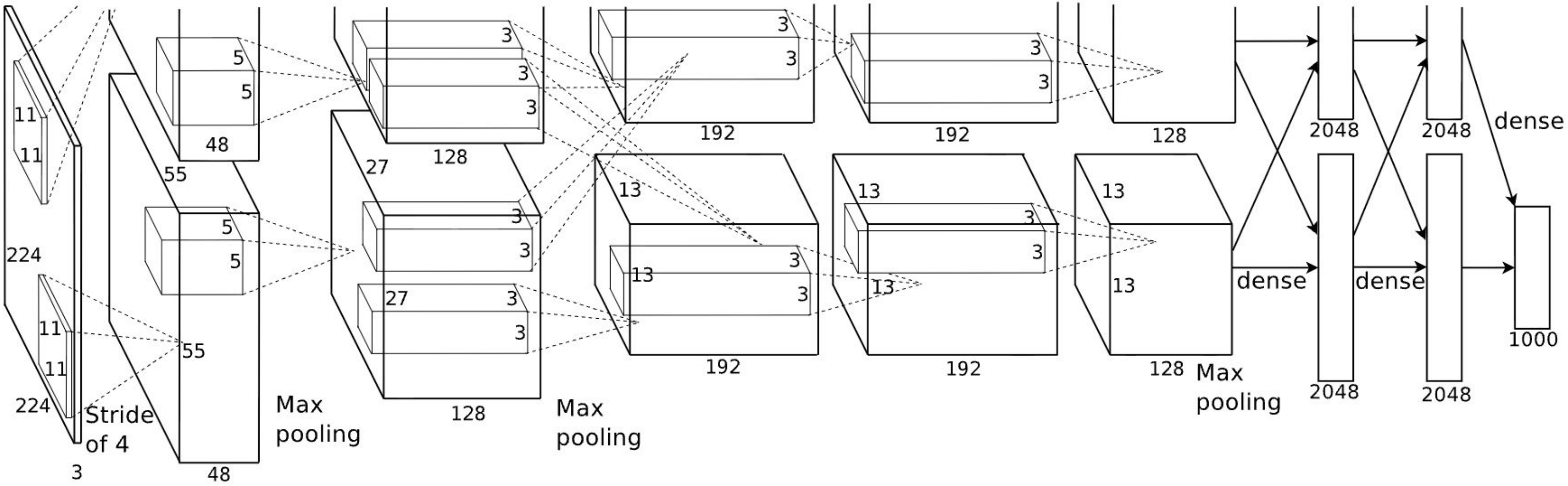
Cornell Bowers C·IS

CNN Layer Output Visualization



Cornell Bowers C·IS

Our goal today



Cornell Bowers C·IS

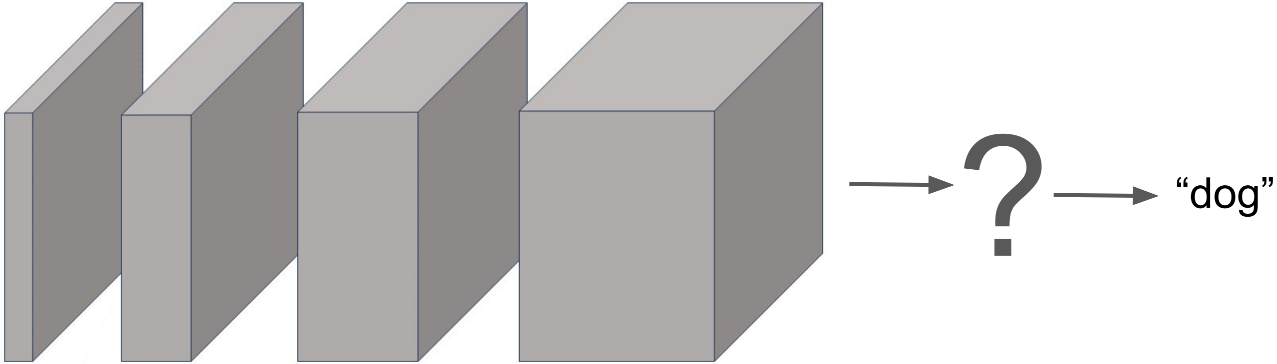
Convolutional Neural Networks (CNNs)

✅ **Convolutions**

Maintain spatial relation between pixels

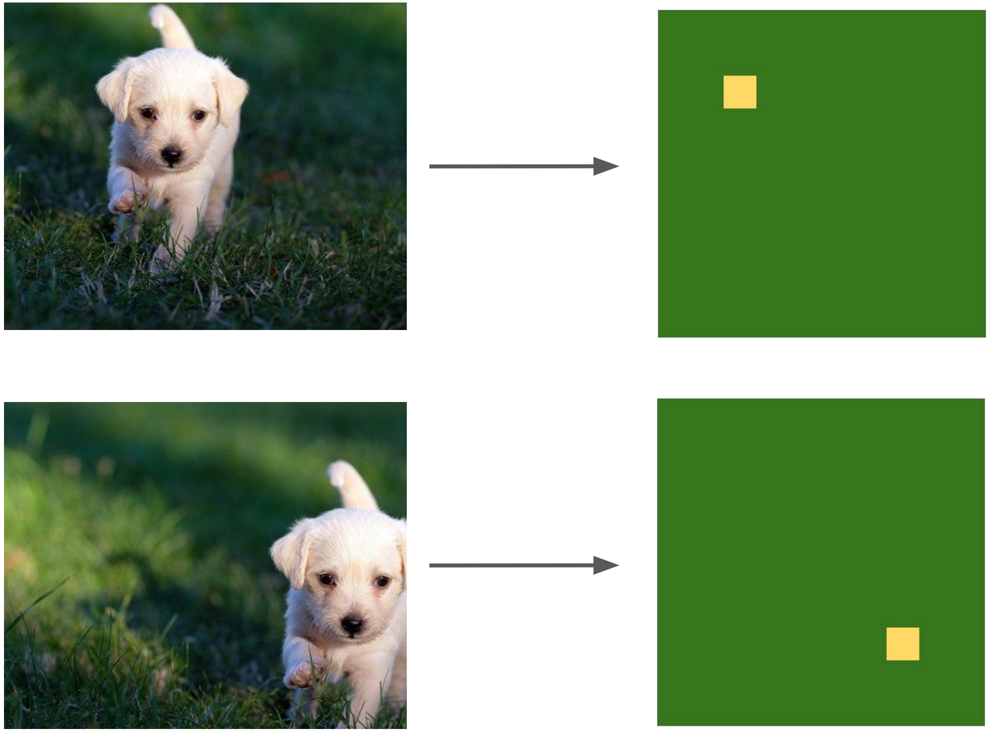
Reduce number of parameters through weight sharing

input image



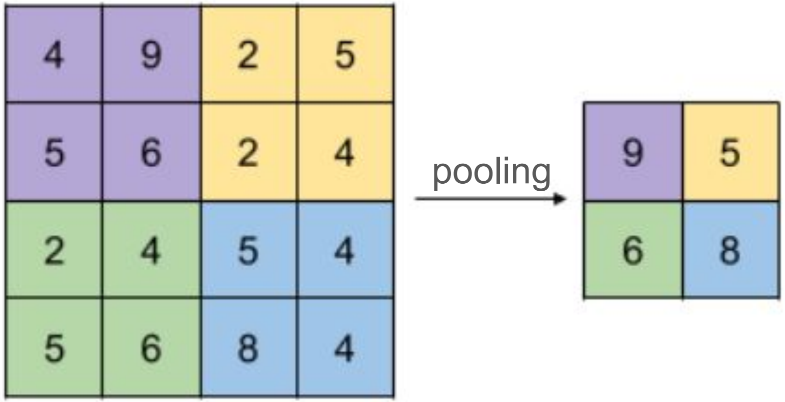
Cornell Bowers C·IS

Ensuring translational invariance



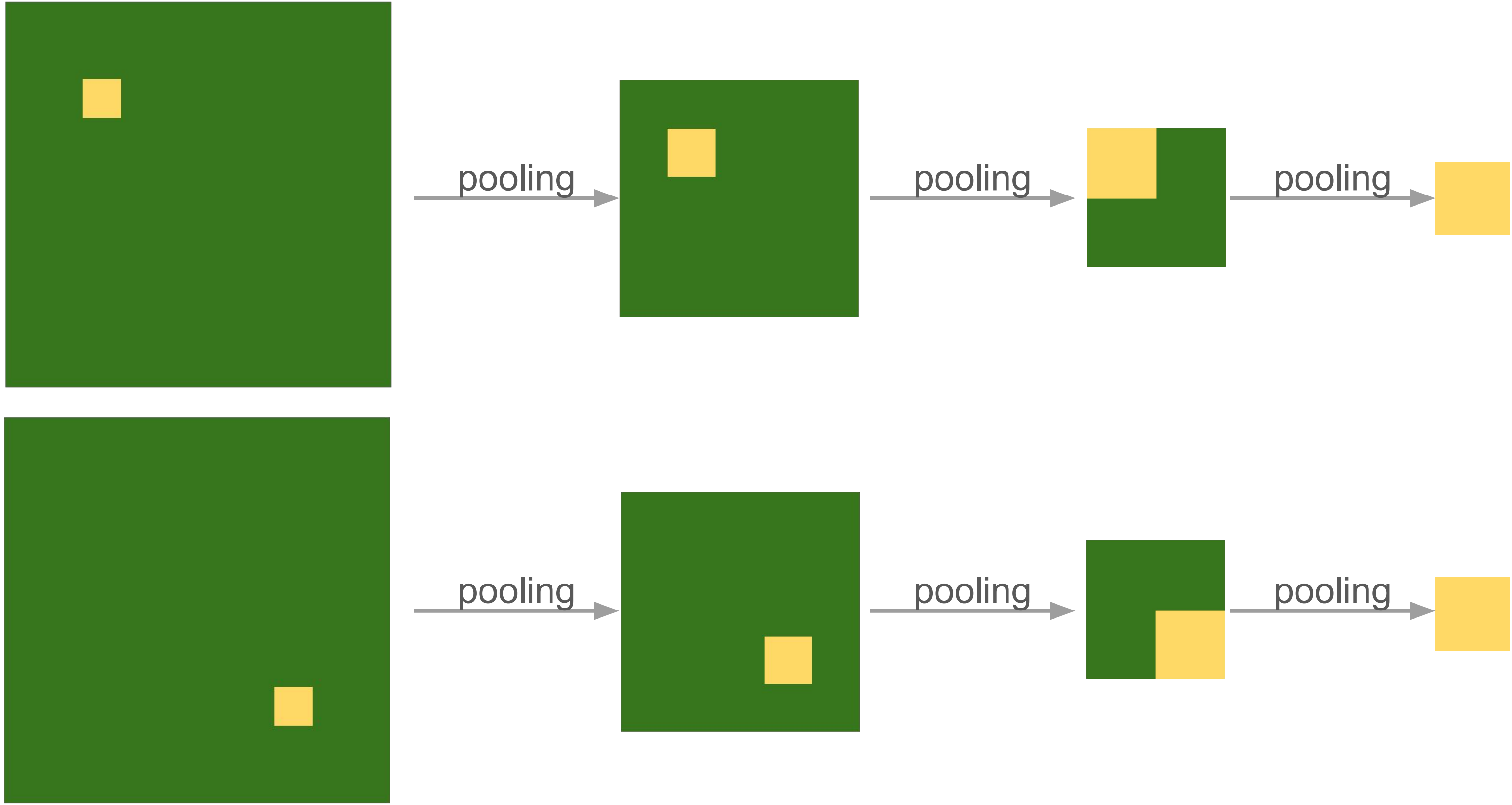
Cornell Bowers C·IS

Max Pooling



Cornell Bowers C·IS

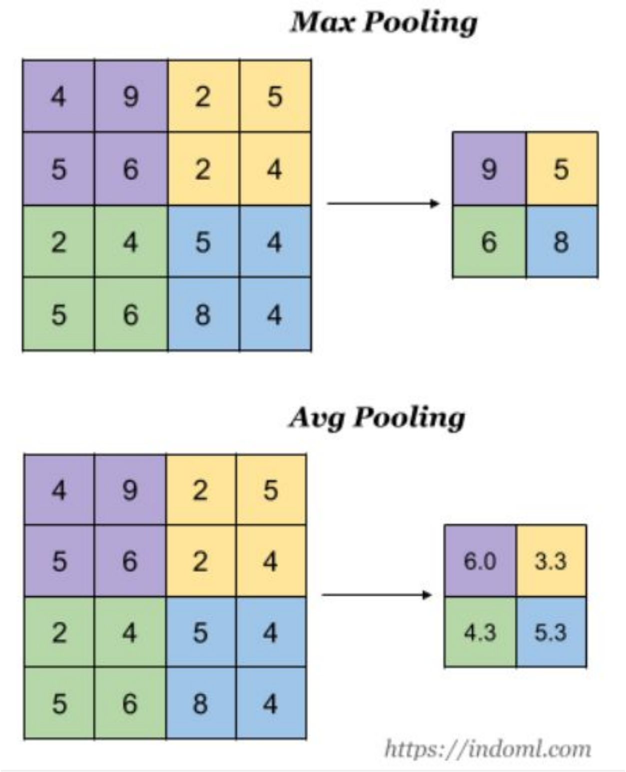
CNNs - Pooling



Cornell Bowers C·IS

CNNs - Pooling

- Down sample feature maps that highlight the most present feature in the patch
- Improve efficiency by reducing computations with downsampling
- Increase receptive field size

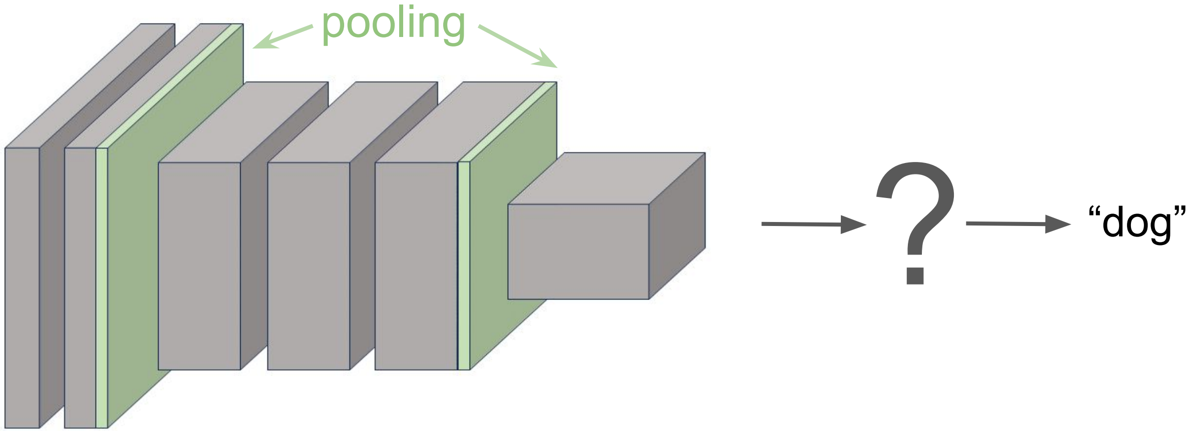


Convolutional Neural Networks (CNNs)

✅ Convolutions — Maintain spatial relation between pixels; Reduce number of parameters through weight sharing

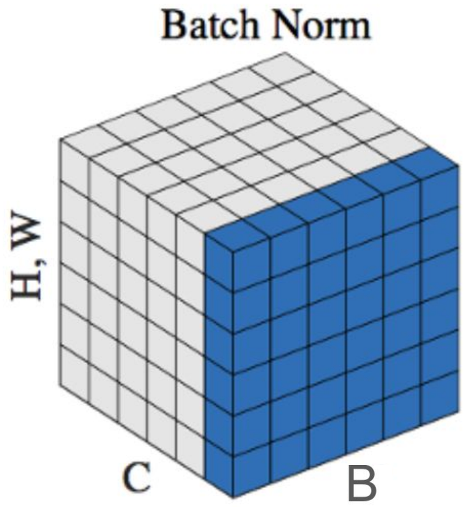
✅ **Pooling** — Captures key information from across different areas of the feature maps; Together with convolutions allows for translational invariance

input image



Normalization

- Normalize channels to mean 0 and variance 1 across each training batch
- Increases speed of training by enabling the use of larger learning rates
- Improves stability of training



Cornell Bowers C·IS

The Batch Normalization Algorithm

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$ // mini-batch mean

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2$$ // mini-batch variance

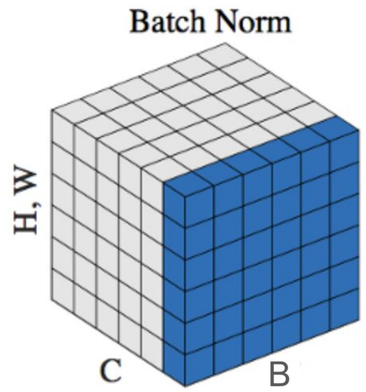
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$ // normalize

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i)$$ // scale and shift

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

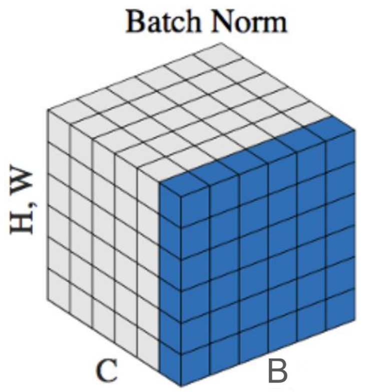
Discuss!

What is the dimension of the mean when you compute the batch norm of a volume of dimension (b x c x h x w)?



Discuss!

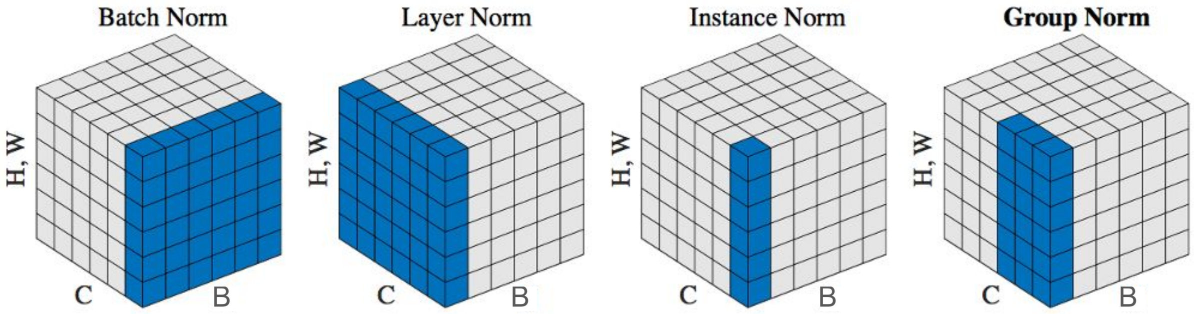
What is the dimension of the mean when you compute the batch norm of a volume of dimension (b x c x h x w)?



The mean should be (1 x c x 1 x 1)!

Normalization Layers

- Normalization layers improve training stability
- Can train with larger learning rates
  - Faster training
- A large learning rate acts as an implicit regularizer
  - Better generalization
- Normalization can also be applied across different dimensions for different use cases



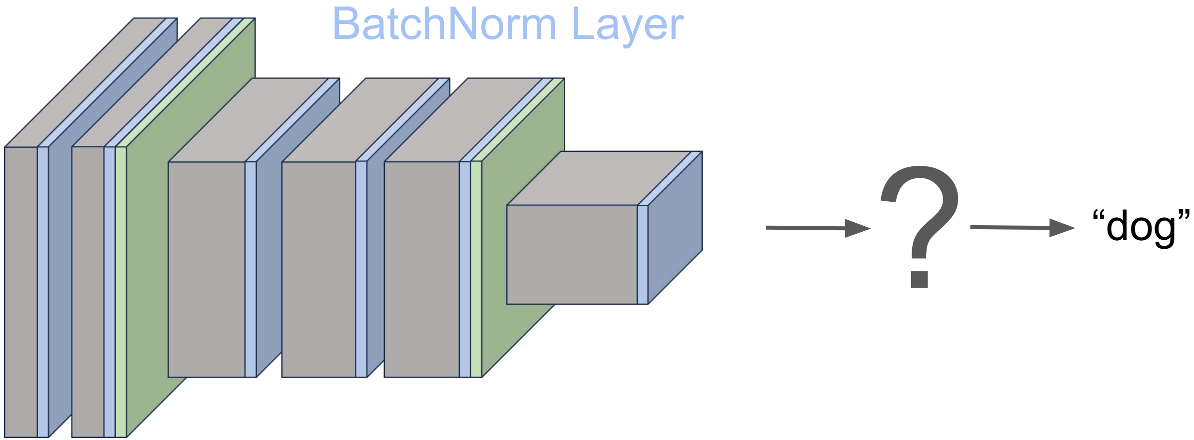
Convolutional Neural Networks (CNNs)

✅ Convolutions — Maintain spatial relation between pixels
Reduce number of parameters through weight sharing

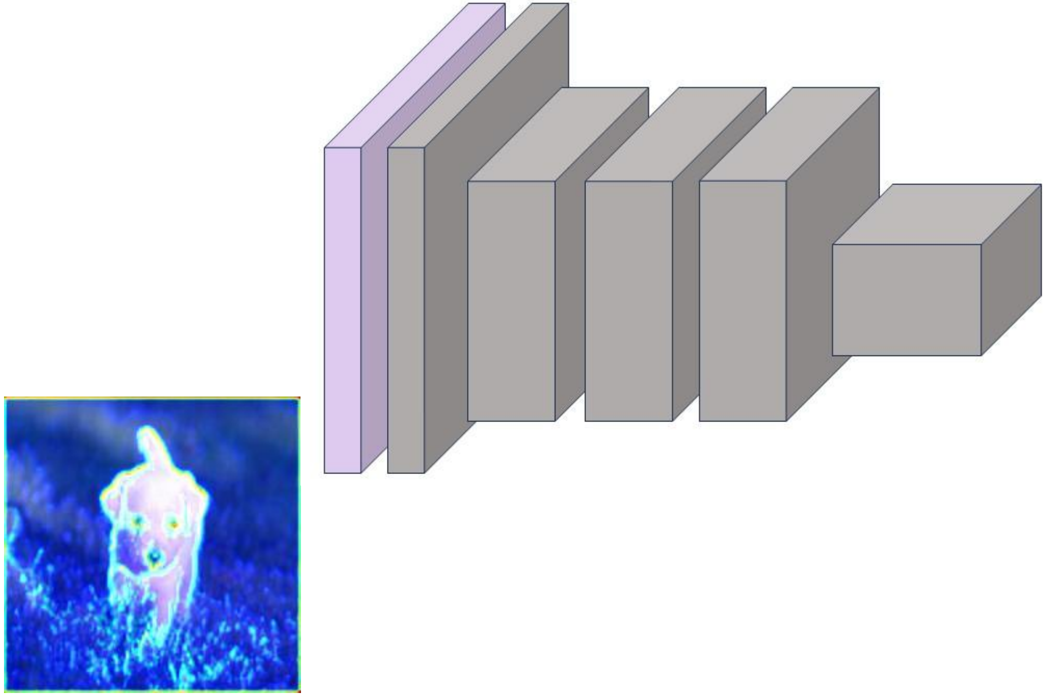
✅ Pooling — Captures key information from across different areas of the feature maps
Together with convolutions allows for translational invariance

✅ **BatchNorm** — Increases speed and stability of training


input image

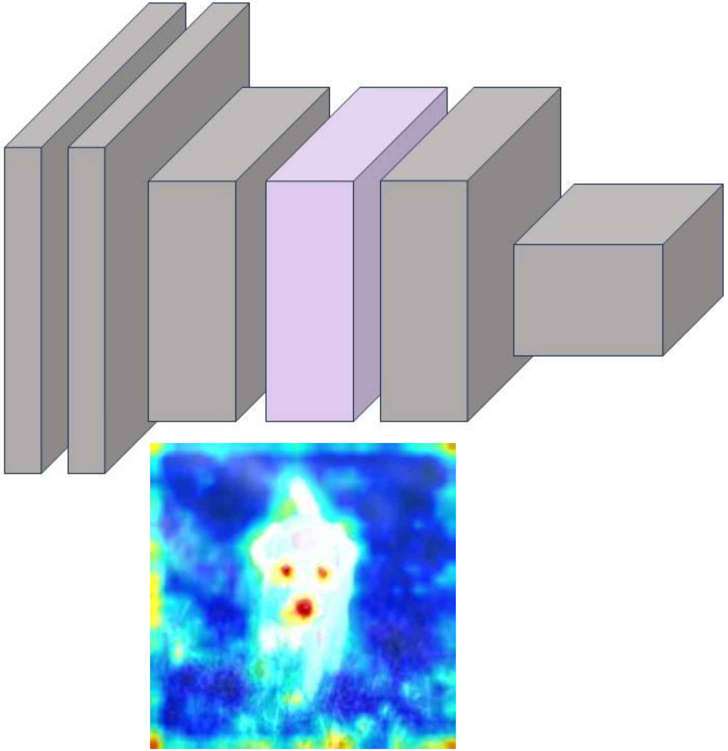


Convolutional Neural Networks (CNNs)



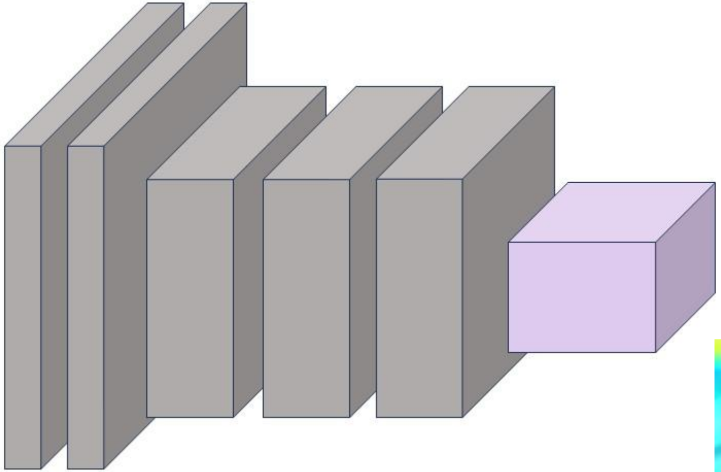
Cornell Bowers C·IS

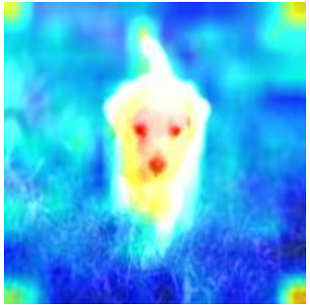
Convolutional Neural Networks (CNNs)



Cornell Bowers C·IS

Convolutional Neural Networks (CNNs)





Cornell Bowers C·IS

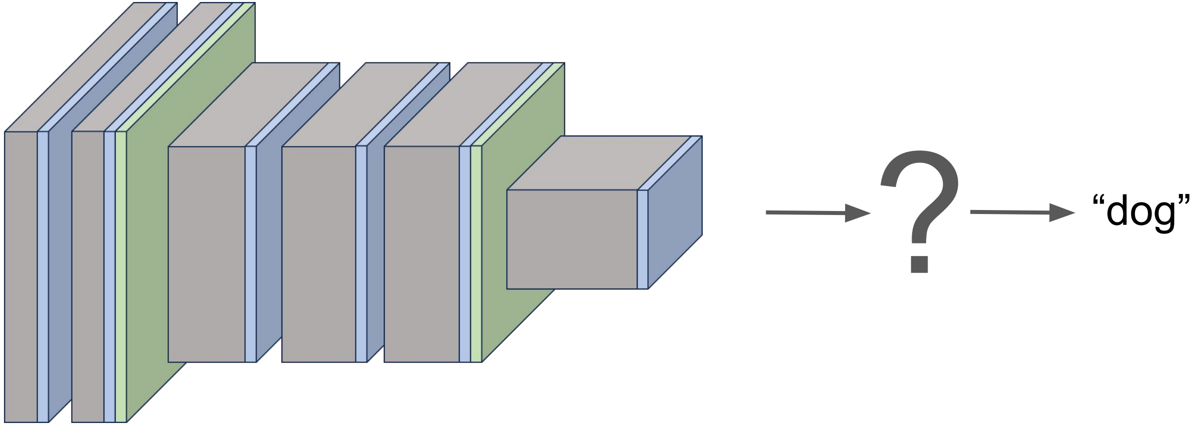
Convolutional Neural Networks (CNNs)

✅ Convolutions — Maintain spatial relation between pixels
Reduce number of parameters through weight sharing

✅ Pooling — Captures key information from across different areas of the feature maps
Together with convolutions allows for translational invariance

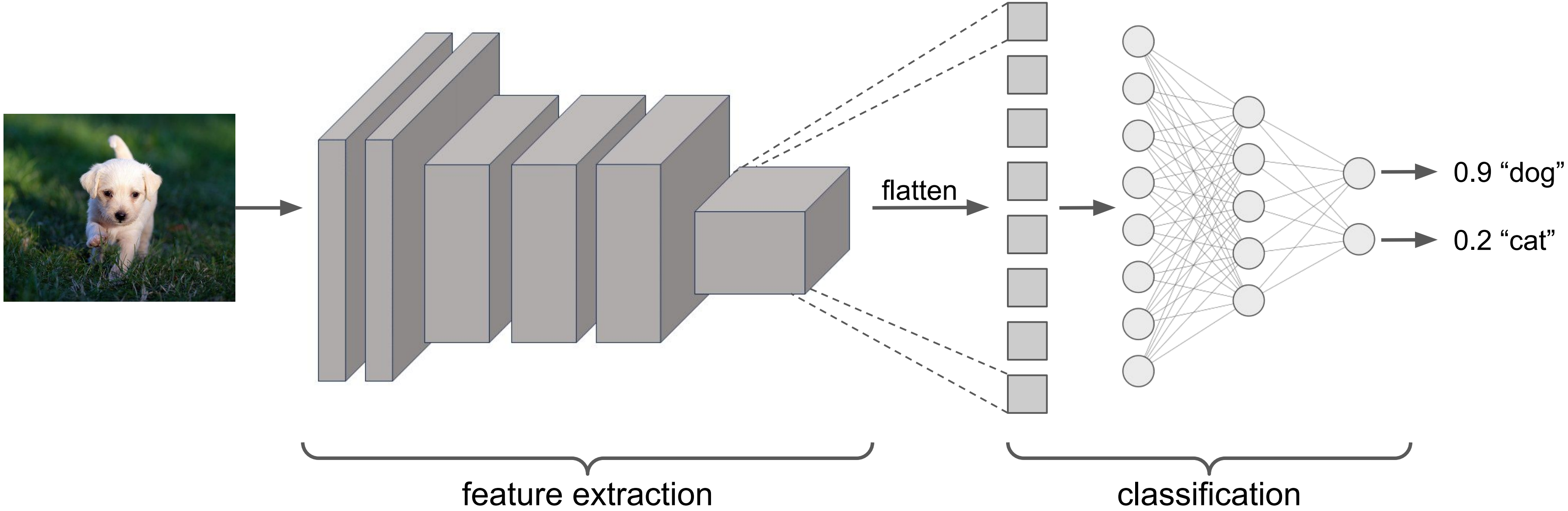
✅ BatchNorm — Increases speed and stability of training

input image



Cornell Bowers C·IS

Image Classification



Cornell Bowers C·IS

Practical Guide

- Input image dimensions is divisible by 2
- Small conv filters (3x3 or 5x5)
- Zero padding is used to maintain spatial resolution
- Max pooling for downsampling
- Pooling layers have a receptive field of 2 and stride of 2

Cornell Bowers C·IS

Summary

- CNNs are primarily designed to process and analyze visual data, such as images and videos.
- Key components: convolution layers, pooling layers, activation functions, normalization layers
- Advantages:
    - Translational Invariance
    - Parameter sharing
    - Feature learning
- Can be trained with backprop
- Used for tasks such as segmentation, classification, object detection, etc.