# Convolutional Neural Networks and Data Augmentation

June course: Deep Learning in Computer Vision

Morten Hannemose, mohan@dtu.dk

June 6th, 2019

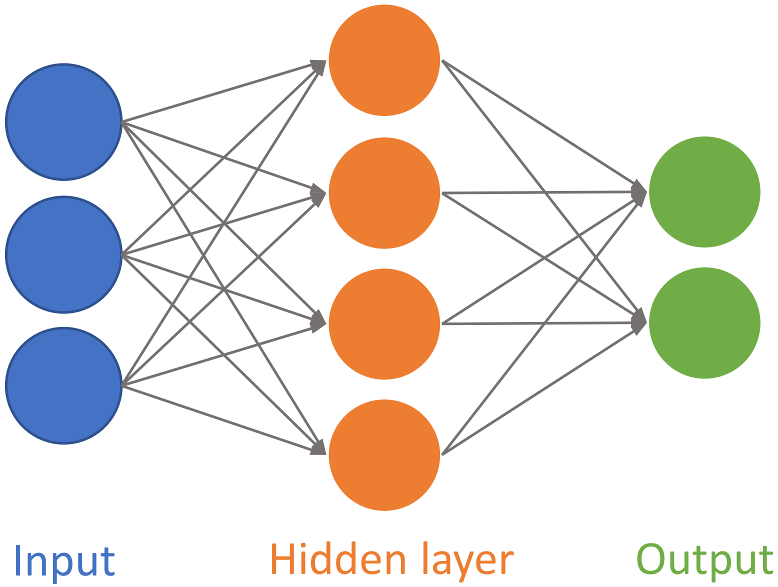
# Outline – What you're going to see

- Convolutions recap
- CNN
    - Convolutions
    - Max pooling
    - Stride/padding
    - Backprop
- Combatting overfitting:
    - Dropout
    - Data augmentation

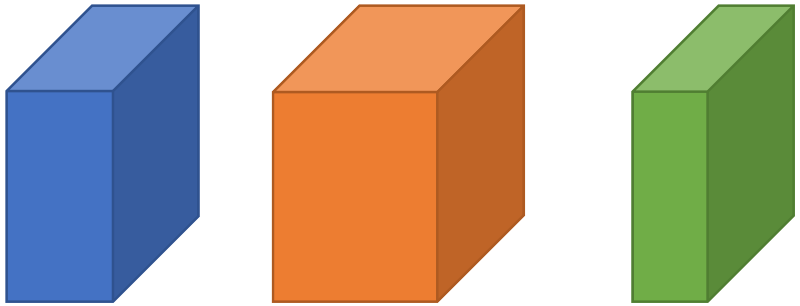
# What is a CNN?

- Convolutional Neural Network
  - Uses convolutions
- Local connectivity
- Weight sharing

Before:
(Fully connected)




After:
(Convolutional)



# Convolutions

- Convolutions and cross correlation are related operations, but convolution involves rotating the filter 180 degrees.
  - We will refer to cross correlation as "convolution" in this course, as is common in deep learning

| | | | | |
|---|---|---|---|---|
| 9 | 1 | 2 | 2 | 7 |
| 10 | 3 | 10 | 5 | 1 |
| 2 | 6 | 10 | 10 | 9 |
| 10 | 10 | 5 | 8 | 10 |
| 7 | 10 | 9 | 10 | 7 |

*

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

=

| | | | | |
|---|---|---|---|---|
| | | | | |
| | -1 | -9 | ? | |
| | -11 | -8 | 6 | |
| | 0 | 0 | -7 | |
| | | | | |

# Convolutions

$$2 \cdot 1 + 2 \cdot 0 \ + 7 \cdot (-1) +$$
$$10 \cdot 2 + 5 \cdot 0 \ + 1 \cdot (-2) +$$
$$10 \cdot 1 + 10 \cdot 0 + 9 \cdot (-1) = \mathbf{14}$$

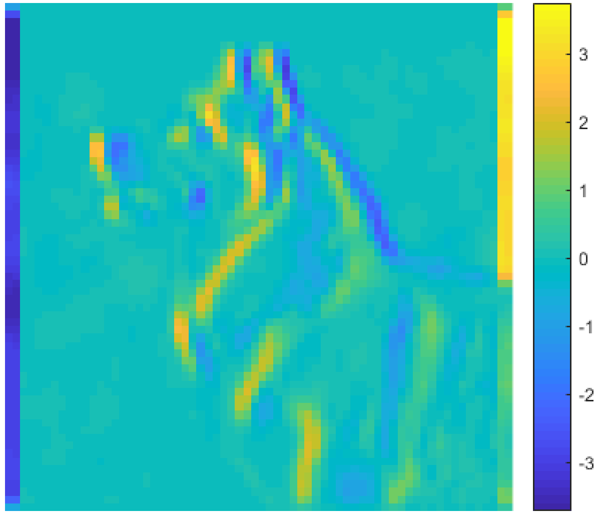
$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 9 & 1 & 2 & 2 & 7 \\ 10 & 3 & 10 & 5 & 1 \\ 2 & 6 & 10 & 10 & 9 \\ 10 & 10 & 5 & 8 & 10 \\ 7 & 10 & 9 & 10 & 7 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} & & & & \\ & -1 & -9 & \mathbf{14} & \\ & -11 & -8 & 6 & \\ & 0 & 0 & -7 & \\ & & & & \end{bmatrix}$$

# Convolutions

- This filter detects vertical edges

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



# Convolutions

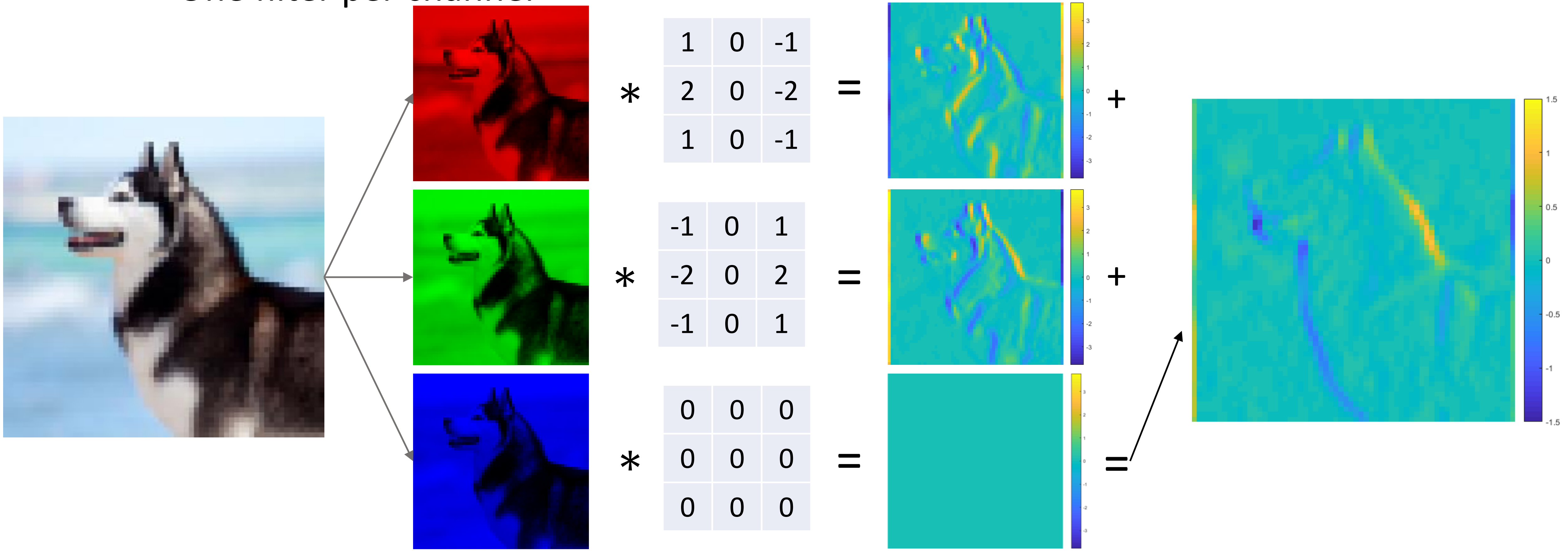
- What do we do when images have multiple channels?

e.g. color images

$$* \begin{matrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{matrix} =$$

# Convolutions

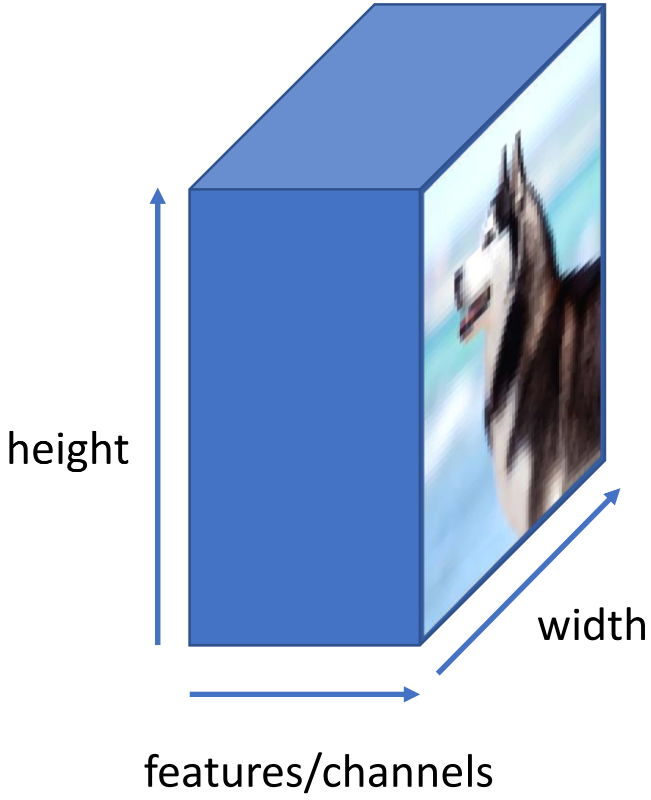
- What do we do when images have multiple channels?
  - One filter per channel



# What is a CNN?

- Exploit 2d layout of images
- Images are volumes

- Color image → three channels
  - Represented in computer as $3 \times 64 \times 64$

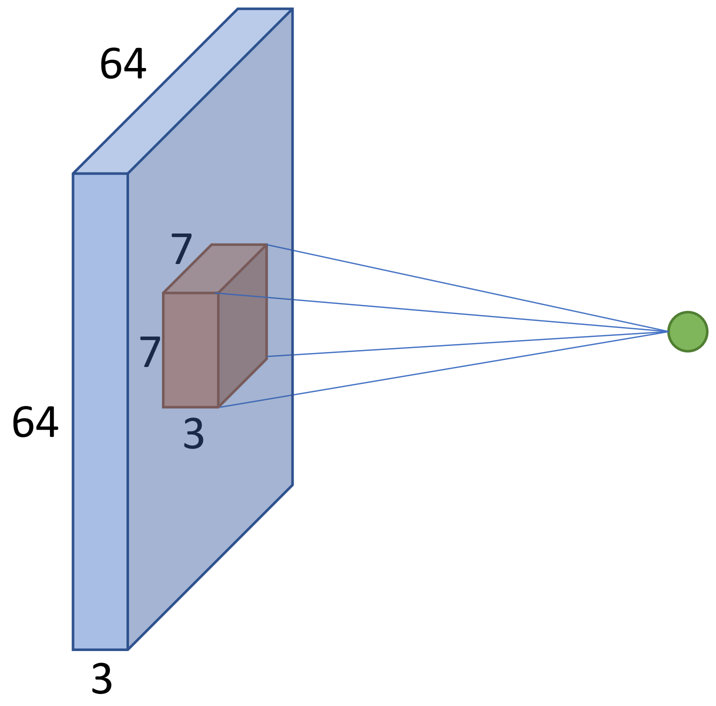


# What is a CNN?

- Let's look at a single neuron
- Input is 3 channels, $64 \times 64$
- $7 \times 7$ convolution

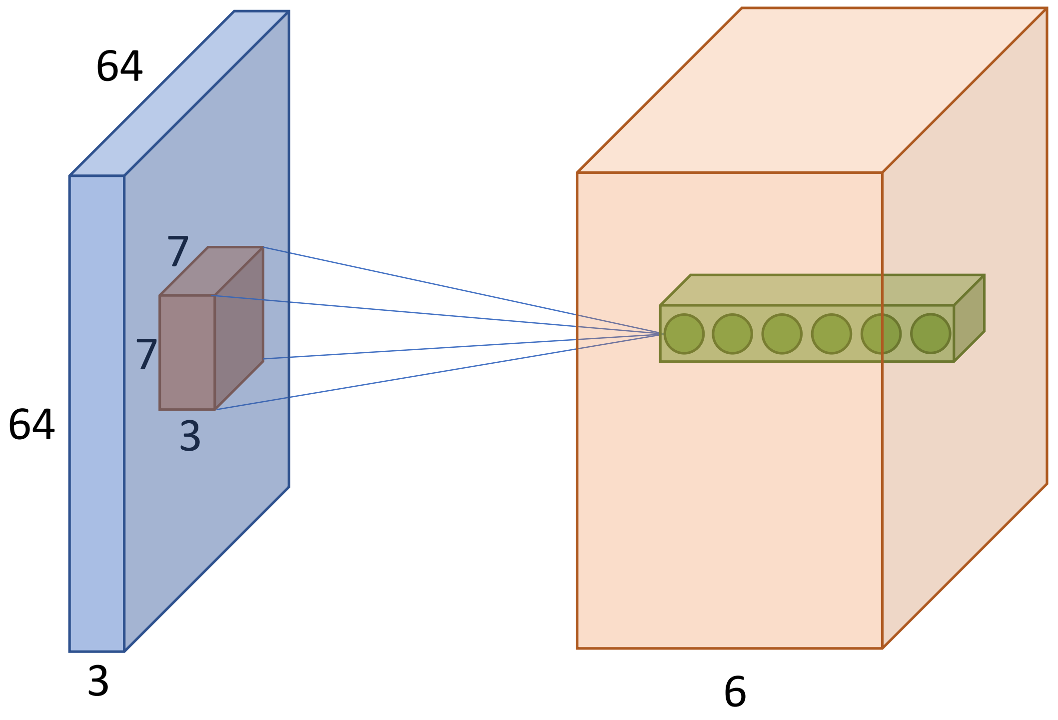
- Each neuron looks at a $3 \times 7 \times 7$ volume in the input layer
    - Requires as many weights + one bias

- Spatially: locally connected
- Depthwise: Fully connected



# What is a CNN?

- Multiple neurons:
- 6 channels (features) output
- Each layer in the output has its own weights
- $6 \times 3 \times 7 \times 7$ weights for this layer



# Mathematical definitions

- Forward pass:

$$z_{i,j}^{\ell,f} = \sum_{c=1}^{C} \sum_{m=-M}^{M} \sum_{m=-N}^{N} w_{m,n}^{\ell,f,c} \cdot a_{i+m,j+n}^{\ell-1,c}$$

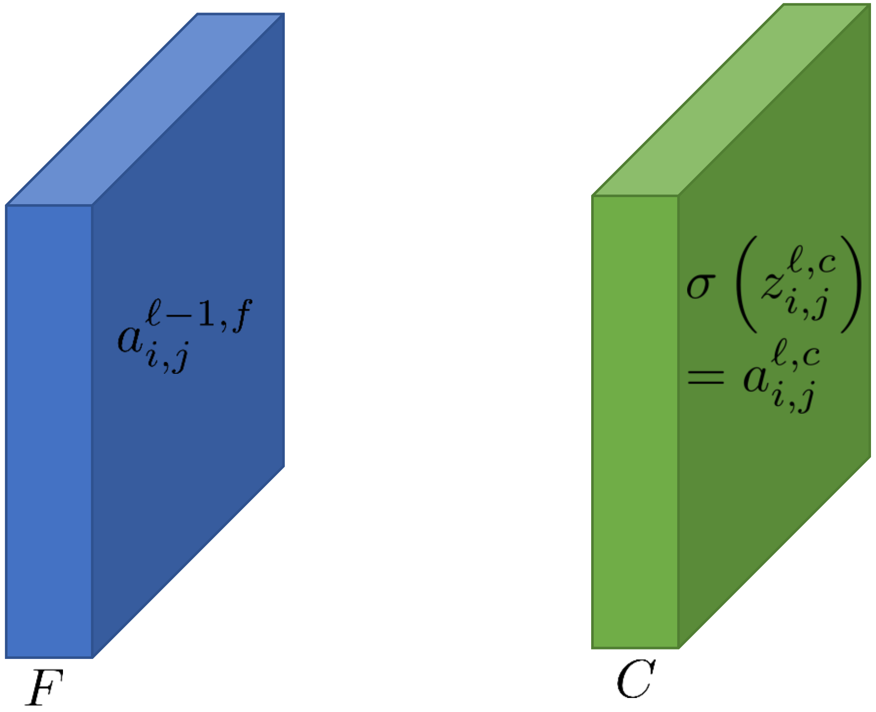
$$a_{i,j}^{\ell,f} = \sigma\left(z_{i,j}^{\ell,f}\right)$$

$$\delta_{i,j}^{\ell,f} = \frac{\partial \mathcal{L}}{\partial z_{i,j}^{\ell,f}}$$

- Backward pass:

$$\delta_{i,j}^{\ell,c} = \frac{\partial L}{\partial z_{i,j}^{\ell,f}} = \sigma'\left(z_{i,j}^{l,c}\right) \sum_{f=1}^{F} \sum_{m=-M}^{M} \sum_{m=-N}^{N} w_{m,n}^{\ell+1,f,c} \cdot \delta_{i-m,j-n}^{\ell+1,c}$$

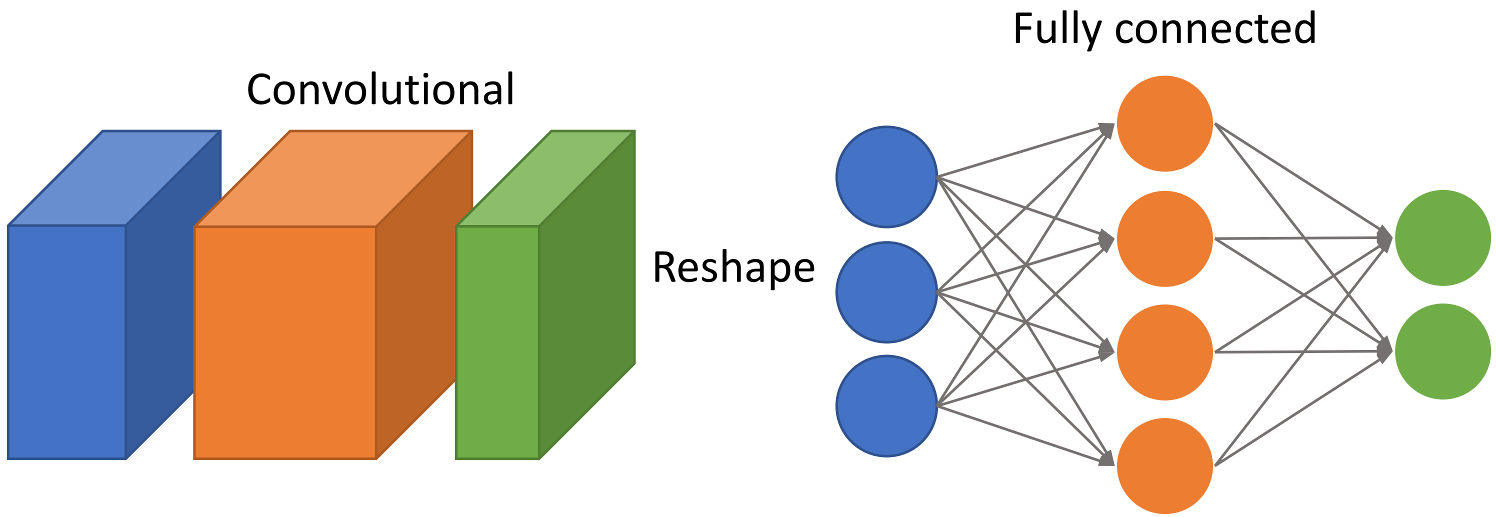
$$\frac{\partial \mathcal{L}}{\partial w_{m,n}^{\ell,f,c}} = \sum_{i} \sum_{j} \delta_{i,j}^{l,f} \cdot a_{i+m,j+n}^{l-1,c}$$



# Wow such image!

# What is a CNN?

- Local connectivity
- Weight sharing
- Usually followed by a fully connected network to output a classification



# Minibatches revisited

- How do we store a minibatch of images in the computer?
  - 4d tensor with size
    - NCHW (minibatch dimension, channels, height, width)
  - Tensorflow has default (which is slower)
    - NHWC

- Minibatches should be made of data sampled without replacement from your full dataset
  - Once all data has been show to the network once it is called an EPOCH
  - After an *epoch* you start sampling all your data over again.

From (2011) Lee et al., Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks

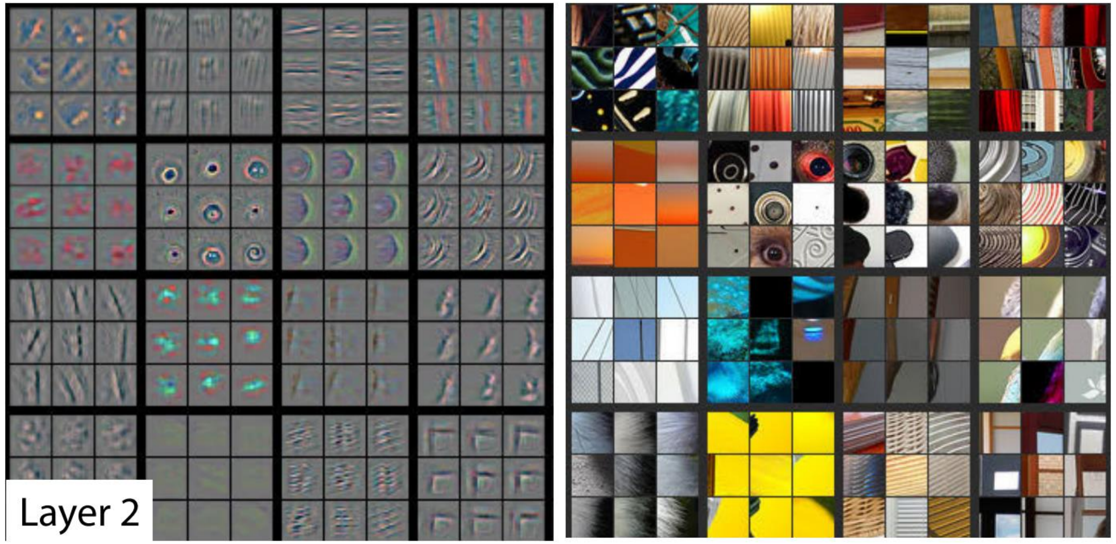
# Intuition

First Layer Representation Second Layer Representation Third Layer Representation

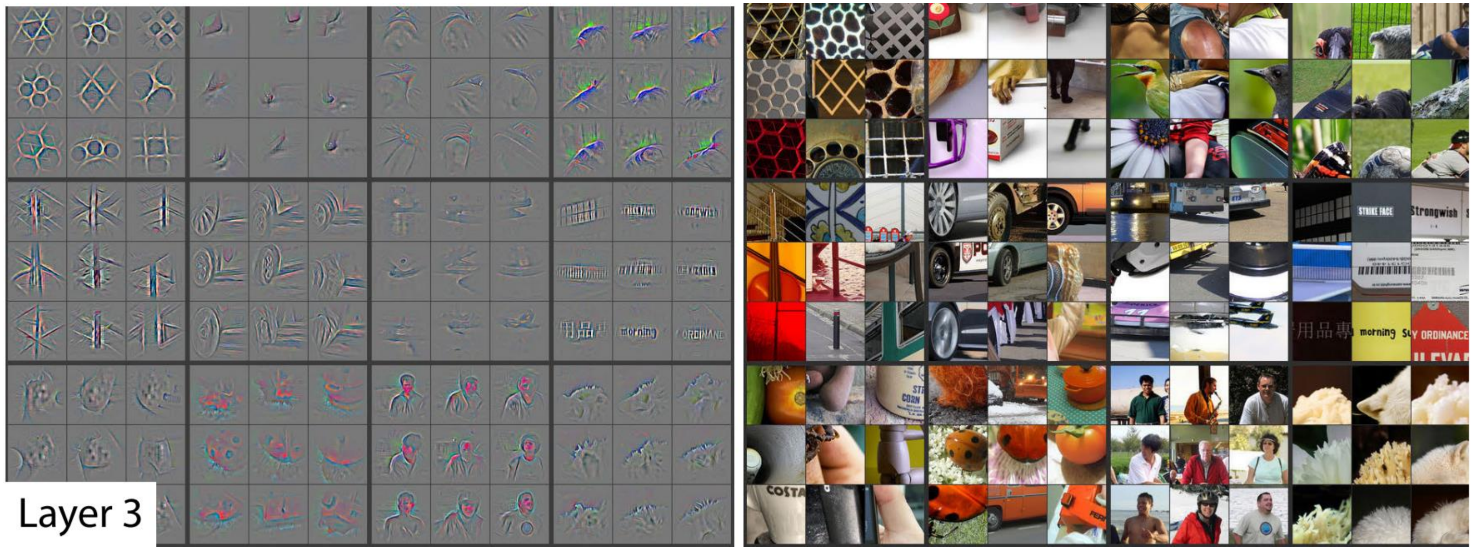
From: (2013) Zeiler and Fergus, Visualizing and Understanding Convolutional Networks
https://arxiv.org/pdf/1311.2901.pdf

# Intuition

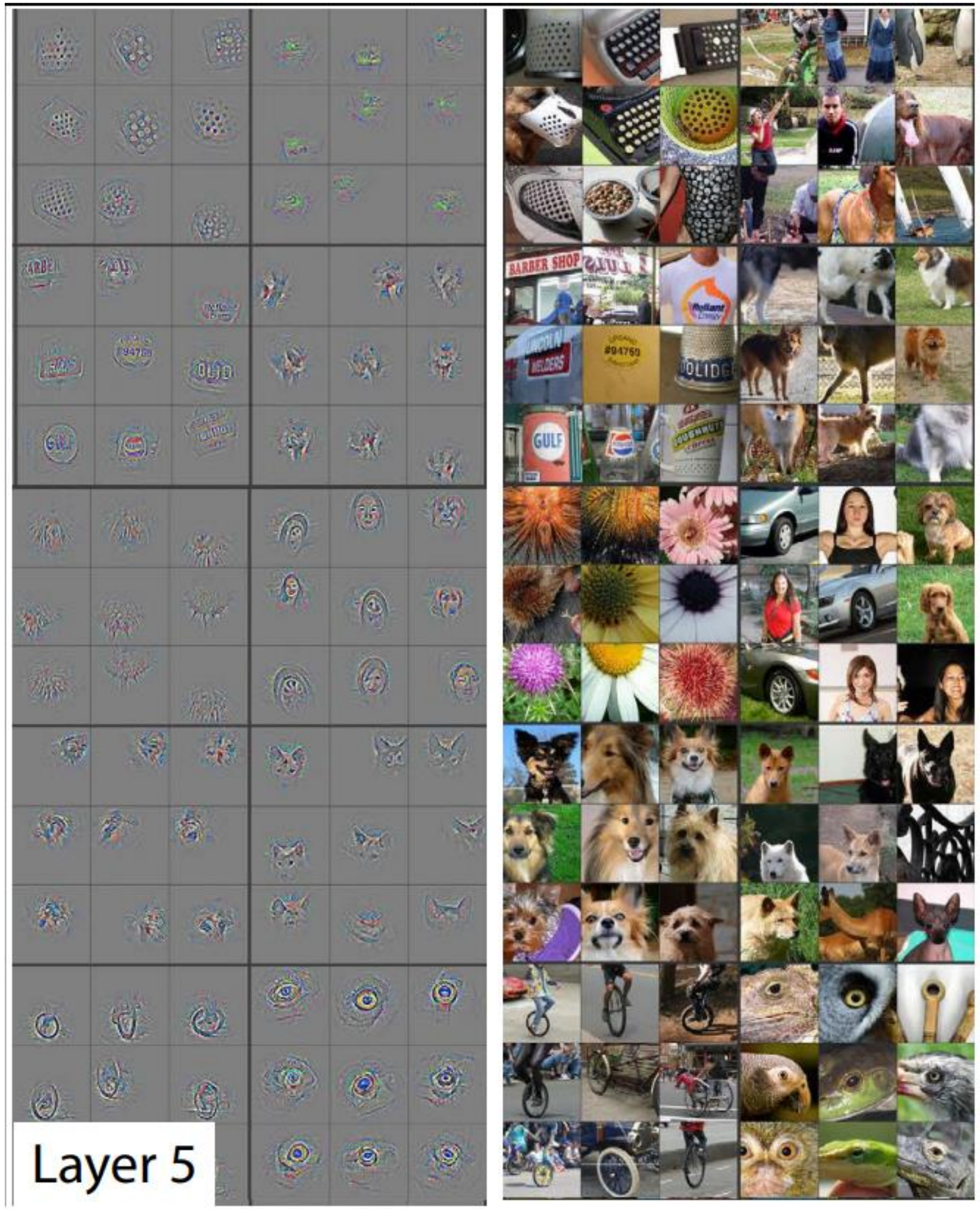
Layer 1



Layer 2



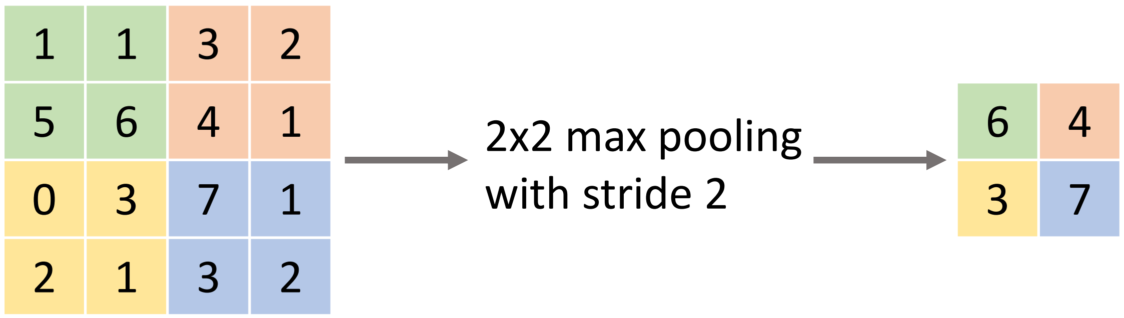
Layer 3



Layer 5

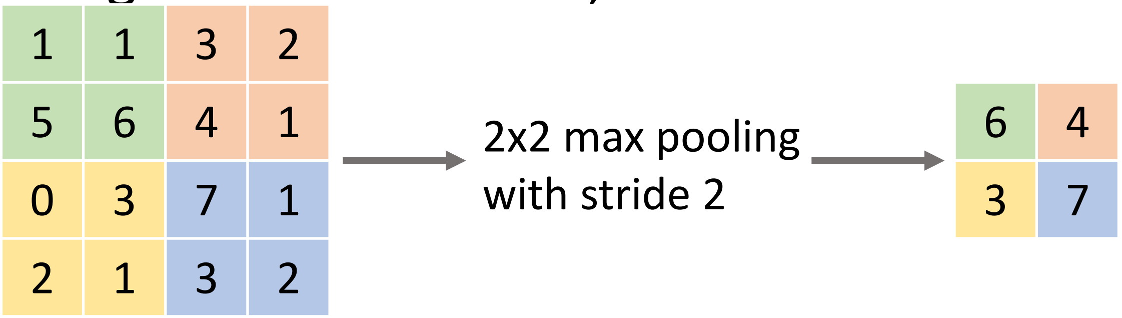
# Max Pooling

- Reducing the spatial size of the feature map
- Example:
  - 2x2 max pooling with stride 2
    - For each 2x2 pixels in each channel, retain only the largest number
    - This type of pooling is extremely common in CNNs.



# Max Pooling

- Pooling reduces the spatial dimension of the features
  - Not the number of channels
- Number of features is reduced by $2 \cdot 2$
  - Makes computation easier
  - A pooling layer is often followed convolution that doubles the number of features.
  - More higher level features, but lower resolution of them: good for classification



# Types of pooling

- Max pooling
  - Max of the values
- Average pooling
  - Mean of the values
- Stochastic pooling
  - A random of the values

- Max pooling is the most common for classification

# Max pooling – most used

- Max pooling is the most common for networks doing classification
  - For classification is does not matter much where exactly a feature is present
  - Taking the largest value helps make the model invariant to small translations

- Almost all poolings are 2x2 with stride=2.
  - Larger generate worse results.

# Strided convolutions

- 3x3 convolution
  Stride = 1



Input size: 9x9
Ouput size: 7x7

# Strided convolutions

- 3x3 convolution
  Stride = 2



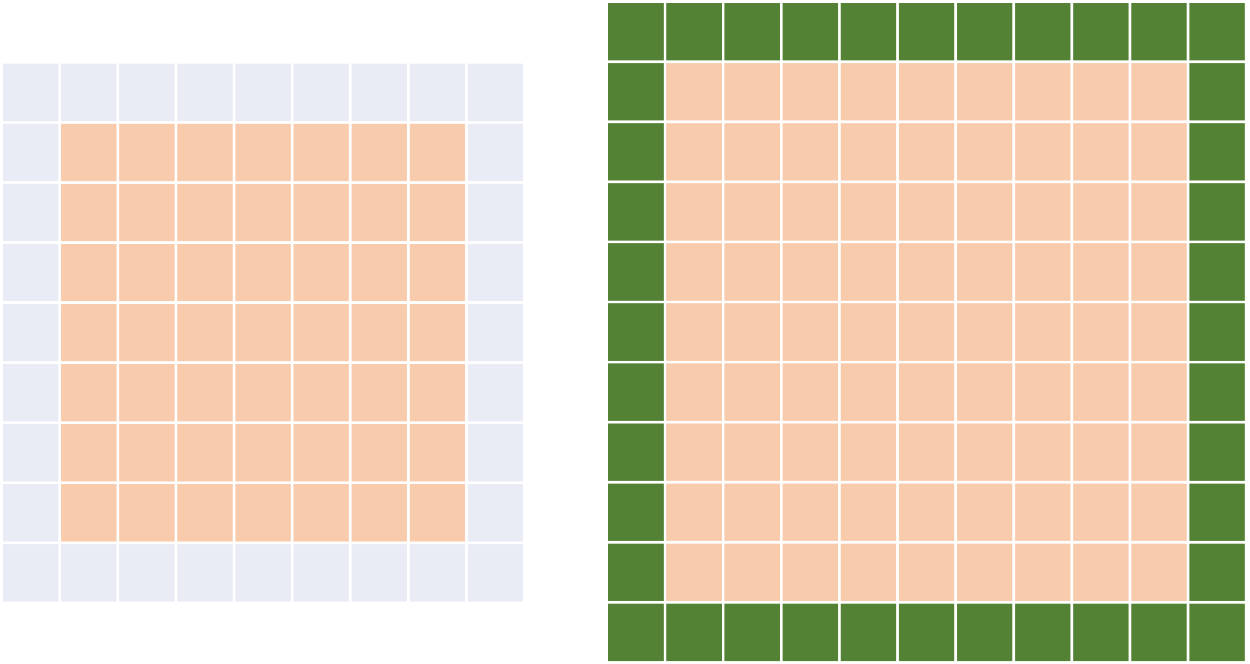
Input size: 9x9
Ouput size: 4x4

# Strided convolutions

- Can reduce the size of the image without using pooling
  - Computationally faster than pooling
    - At the risk of being less accurate
  - Perferred in some GANs due to consistent flow of gradients

# Padding

- The image size is reduced even for stride=1.
- Artificially increase the size of the image before convolution.
- Padding=1 will increase the size by one at the top, bottom, left and right.
- Usually zeros are used as the padding value in CNNs.



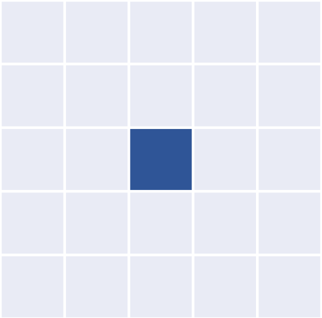
# Padding

- Padding with zeros makes sense when the network contains feature maps
  - Positive values indicate the feature is present
  - Zero means this neuron was not activated (the activation was negative and ReLU made it zero)

- If the size needs to be kept constant, you need to pad with:

$$\frac{n-1}{2}$$

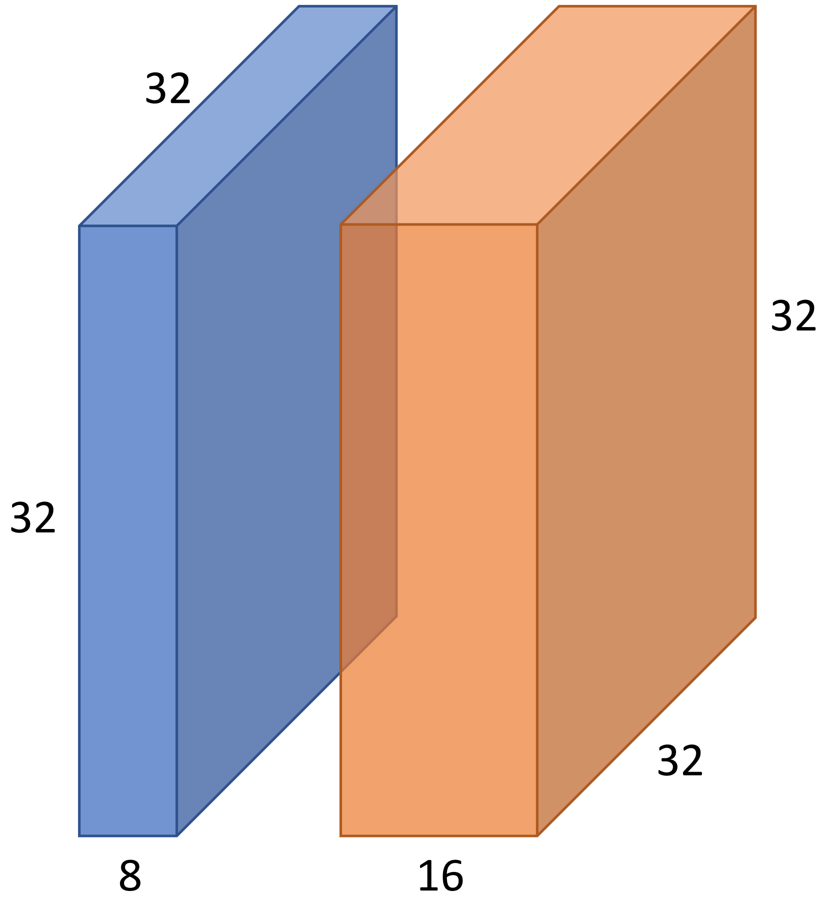
- Visualize the convolution kernel and see how much it has on each side
  - 5x5 → padding=2



# Question

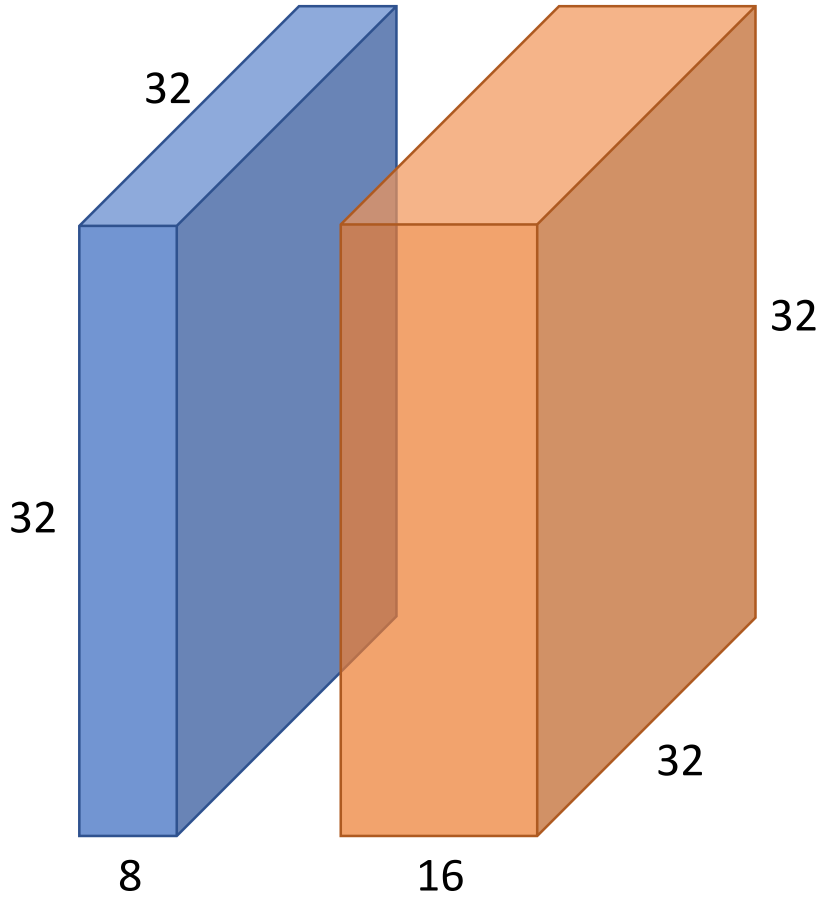
- **DISCUSS WITH YOUR NEIGHBOUR 2 MINUTES**
- Input has 8 channels, and spatial dimensions 32x32
- We perform a 7x7 convolution that produces a new volume with 16 channels and still 32x32 spatially.

- What is the stride?
- What is the padding?
- How many weights (learnable parameters) does the convolution have?



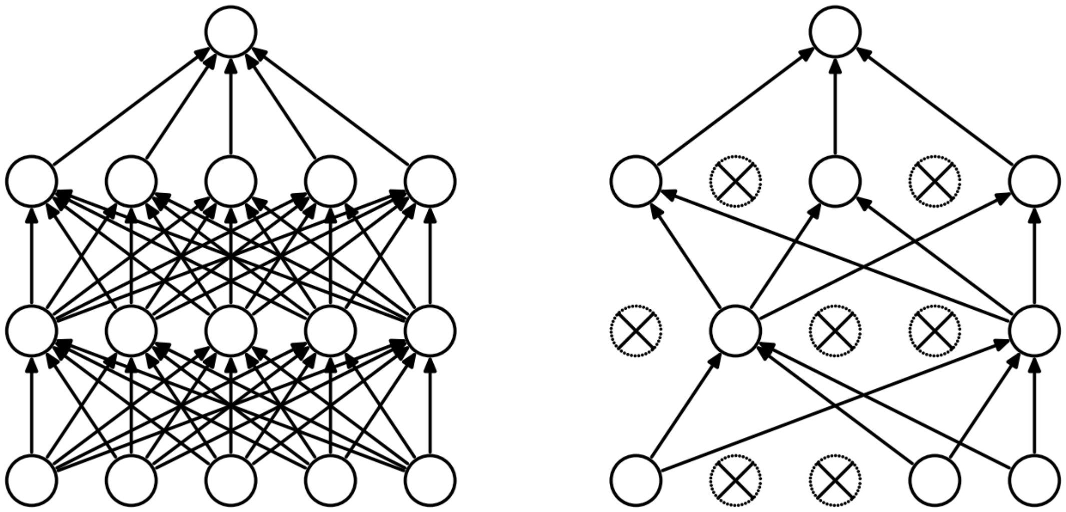
# Question

- What is the stride?
  - Stride=1
- What is the padding?
  - Padding=3
- How many weights (learnable parameters) does the convolution have?
  - 16x8x7x7 + 16 bias



Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting

# Dropout



(a) Standard Neural Net (b) After applying dropout.

- Background
  - Nerual networks have many weights and can easily **overfit** to your data
- Concept: Model ensembles (averages of many) are always good
  - How can we do this in a single model?
- How it works
  - Each forward pass, we randomly omit each feature with a probability of 0.5
  - This means we are actually sampling from $2^n$ different architectures
  - Efficient way of performing model averaging with neural networks

# Dropout

- Imagine if you each day only had either your right or left arm.
  - You would be forced to become good at using both arms, because you don’t know which arm you will have tomorrow.

- Intuition:
  - Specialized neuron might be good enough to classify correctly
    - Other neurons become lazy
  - Randomly removing neurons forces all neurons to do their best

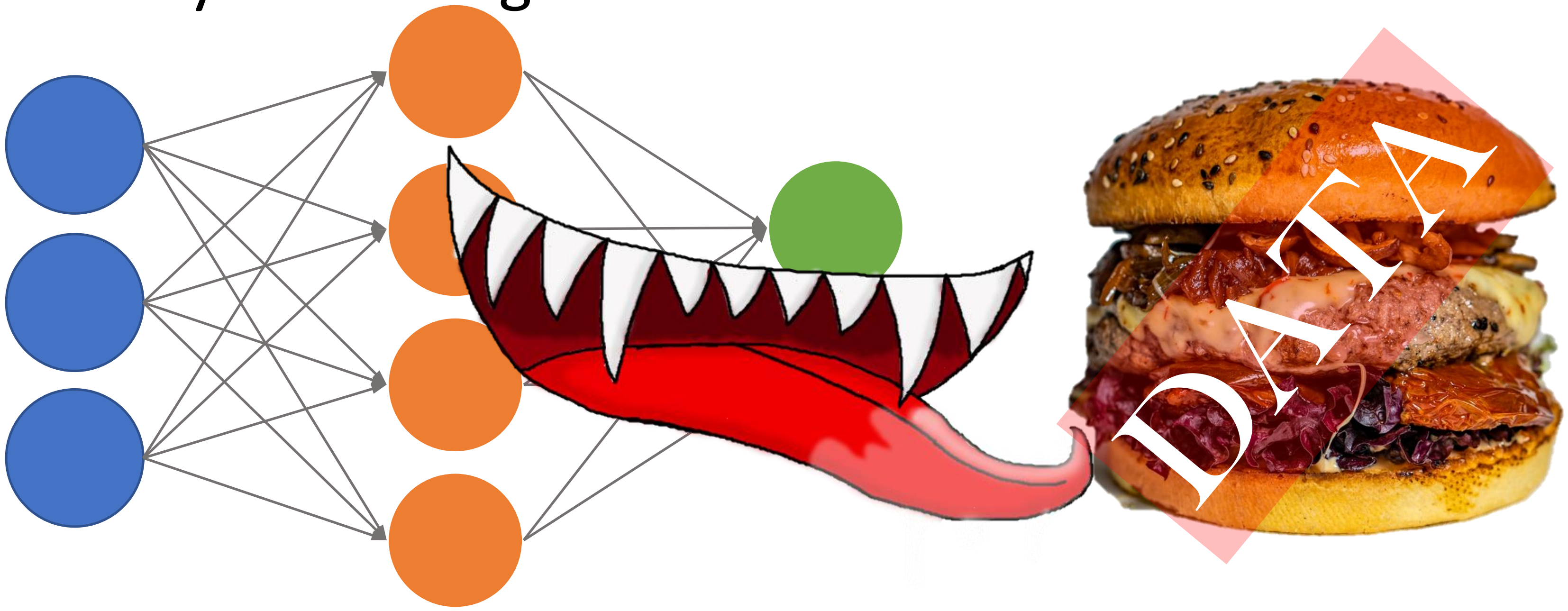
# Dropout

- Technical details:
  - The dropped neurons during training means expectation of layer output is smaller than during training
    - This is problematic
  - It is common to scale the activations up by the dropout factor during training
    - For example if we drop p=50% of neurons, during training we multiply the activations by 1/p=2
- This is often handled by the high-level framework you are using
  - As long as you tell the model whether it's training or testing right now.

- Dropout is a well known regularization technique, but BatchNormalization is an often used alternative.

# Data augmentation

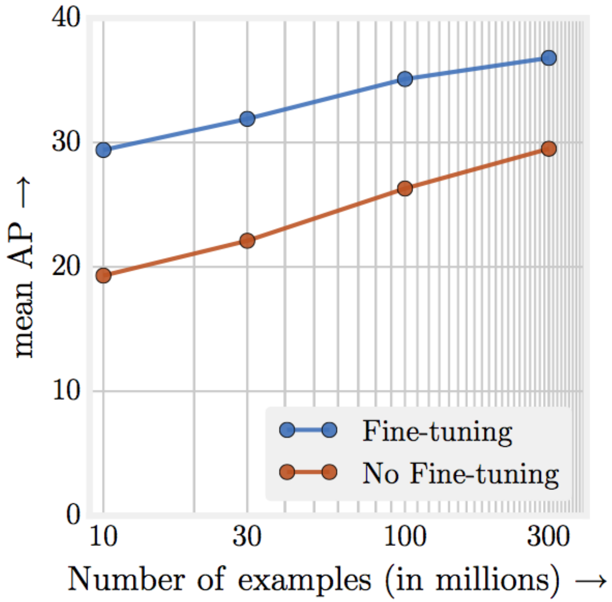
- Neural networks are very data-hungry
- How to satisfy their hunger?



# Data augmentation

- Neural networks are very data-hungry
- How to satisfy their hunger?

- "just have enough data"
  - ImageNet has 14 million images
    - not big enough
  - Google has JFT-300M with 300 million images
    - not big enough
    - also not public
- We need to fully utilize the data we do have.




From: https://ai.googleblog.com/2017/07/revisiting-unreasonable-effectiveness.html

# Data augmentation

- What is this?
- Any ideas on how we can create more images?

# Data augmentation

- Create more data by exploiting things that the label is invariant towards
- Always keep in mind what the images are of
  - Flipping an image of text means that the text is no longer readable

- Types of augmentation:
  - Flips
  - Translation
  - Scale (non-uniform?)
  - Rotation
  - Elastic deformation
  - Lens distortion?
  - Noise
  - Blur
  - Small color changes

# What you have learned

- Convolutions recap
- CNN
    - Convolutions
    - Max pooling
    - Stride/padding
    - Backprop
- Combatting overfitting:
    - Dropout
    - Data augmentation

Slides slightly inspired by: Fei-Fei Li & Andrej Karpathy, CS231n:
Convolutional Neural Networks for Visual Recognition, Winter 2015. (Stanford University)