CS167: Machine Learning

Convolutional Neural Network (CNN) Implementation
Popular CNNs

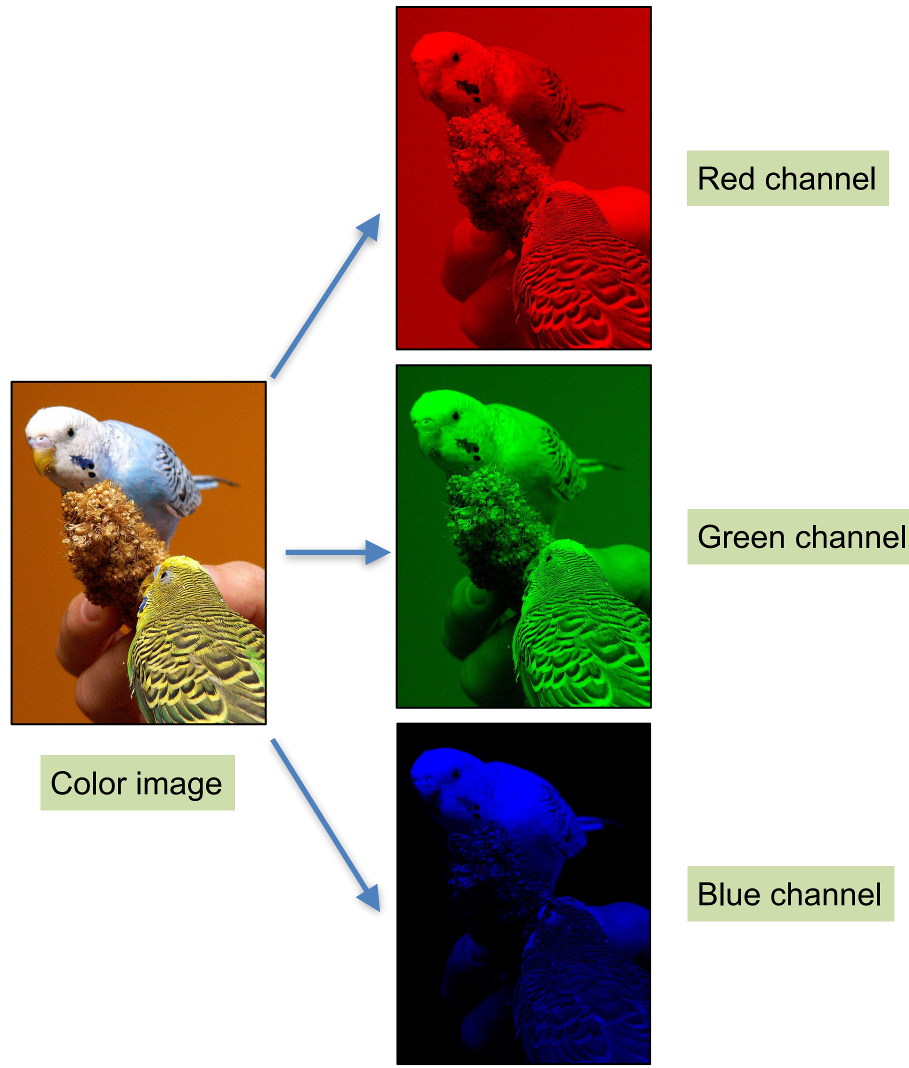
Thursday, April 18th, 2024

Drake UNIVERSITY

Recap: Convolutional Neural Network (CNN)

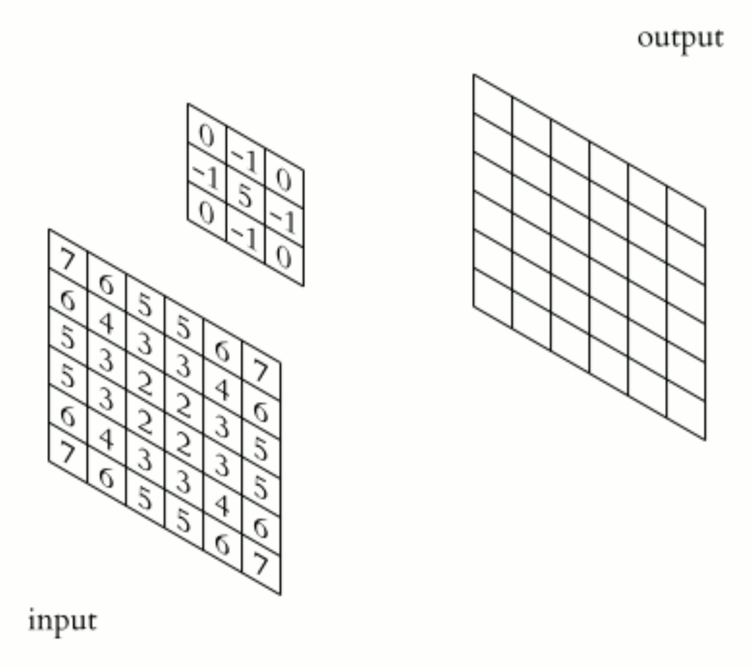
- A **convolutional neural network** that applies **convolutional filters** on grid-like input such as a image

- Image data is represented as a two-dimensional grid of pixels, either grayscale (monochromatic) or color (RBG)
  - each pixel corresponds to one or multiple numeric values: if it's grayscale, it is one number, if it's color, it corresponds to 3 numbers (a red, a green and a blue value)



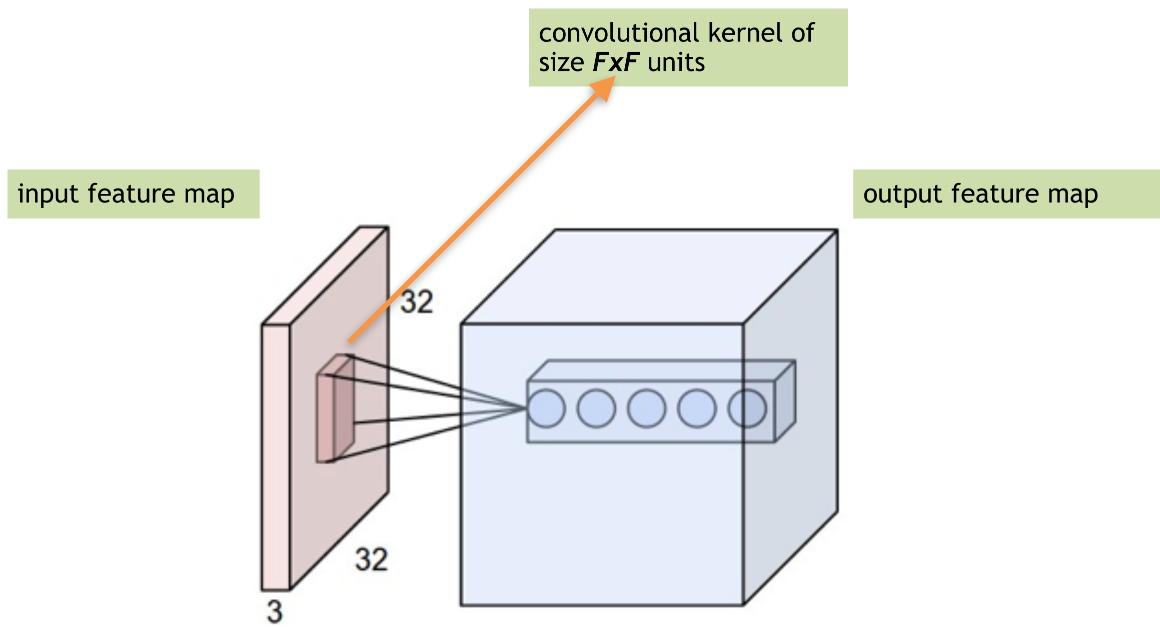
Recap: Convolution Operation

- What does a **convolution operation** do?
- **convolution operation can be achieved with a series of dot products between portions of input feature map and a convolution filter (kernel) weights**



Another visualization shows a convolution filter applied to an image, resulting in the convolved feature

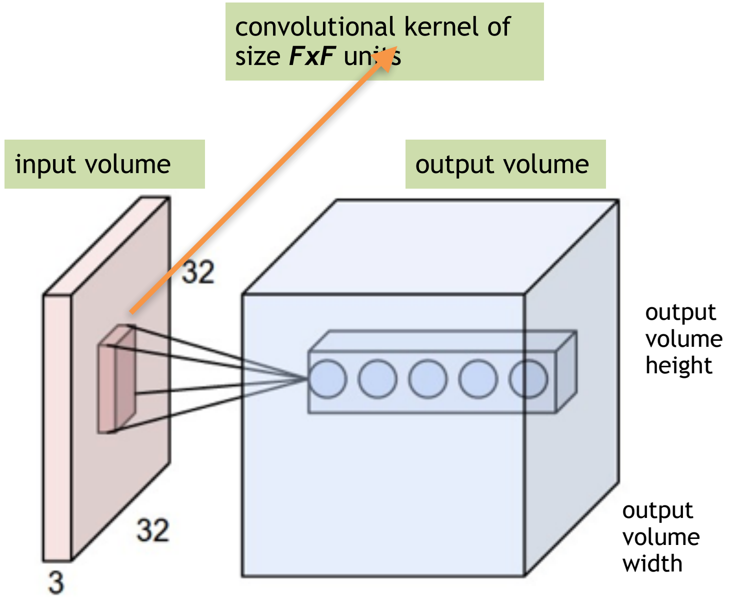
Recap: Convolutional Neural Network (CNN)



- Weights correspond to the filter (kernel) values
- **Convolutional neural network** can **learn** their own filters!
    - We do not need to provide the values inside the kernel

Recap: How to calculate the output volume size?

- An input volume has size An input volume has size **(W_1 x H_1 x D_1)**
  - Filter size/receptive field is ***(FxF)***
  - Spatial stride size ***S***
  - Padding size ***P***
  - Number of filters ***K***



- Spatial sizes of the output volume ***(W_2 x H_2 x D_2)***

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1$$

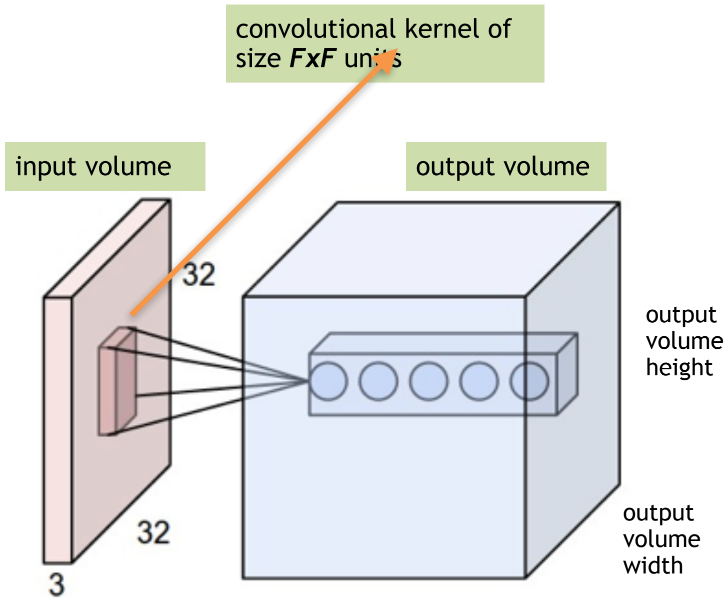
$$H_2 = \frac{(H_1 - F + 2P)}{S} + 1$$

$$D_2 = K$$

- Number of filter weight parameters **= *(F x F x D_1) x K***
- Number of bias parameters = ***K***

Recap: How to calculate the output volume size?

- An input volume has size ***(WxWx3)***, eg, (227, 227, 3)
- Filter size/receptive field is ***(FxF)***, eg, (11x11)
- Spatial Stride ***S***, eg, ***S***=4
- Padding size ***P***, eg, ***P***=0
- Number of filters ***K***, eg, ***K***=96



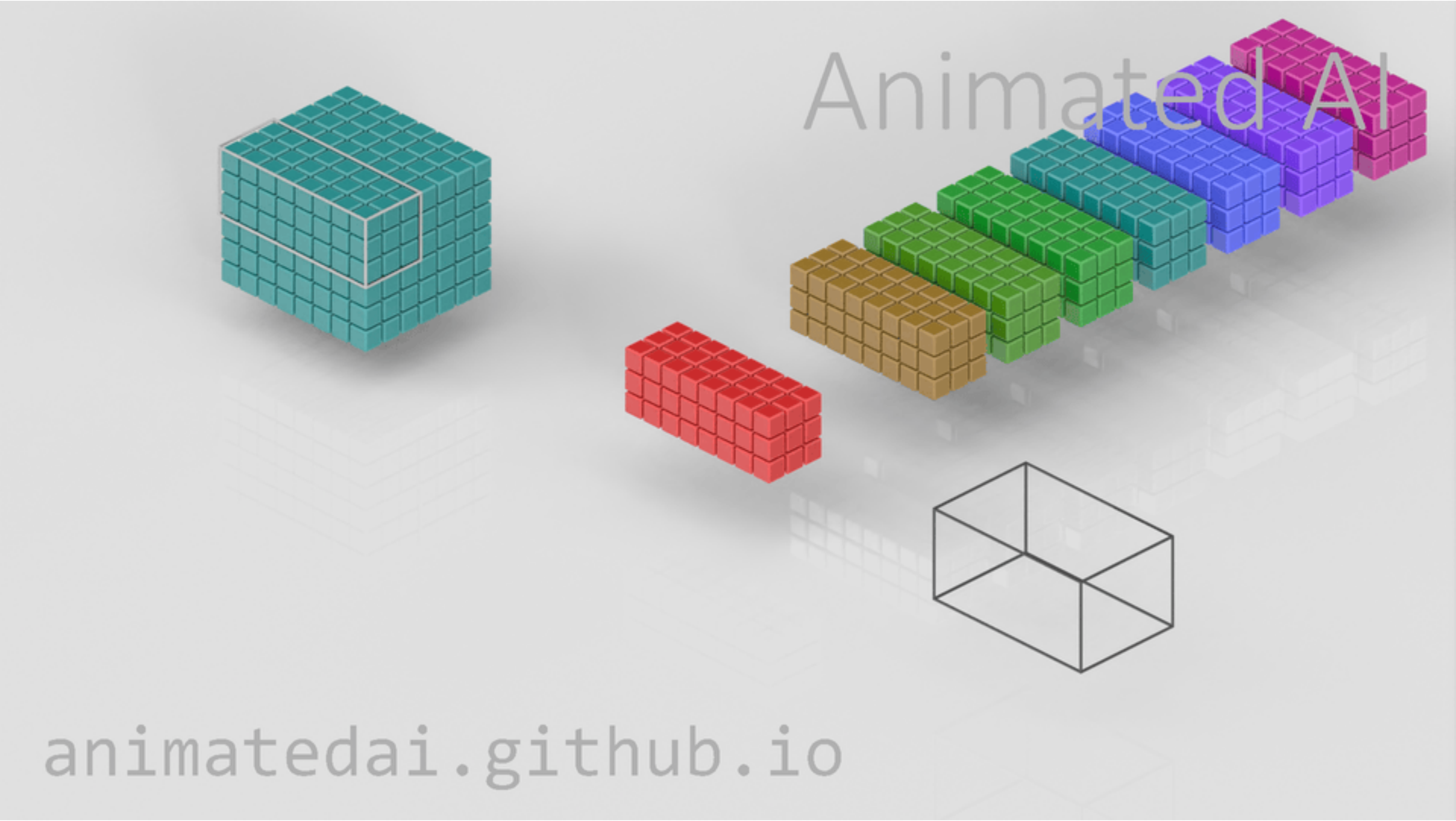
- Size of the output volume width and output volume height as a function of ***W, F, S,***and ***P*** as follows:

$$\text{output volume width/height} = \boxed{\frac{(W - F + 2P)}{S} + 1} = \frac{(227 - 11 + 2*0)}{4} + 1 = 54+1 = 55$$

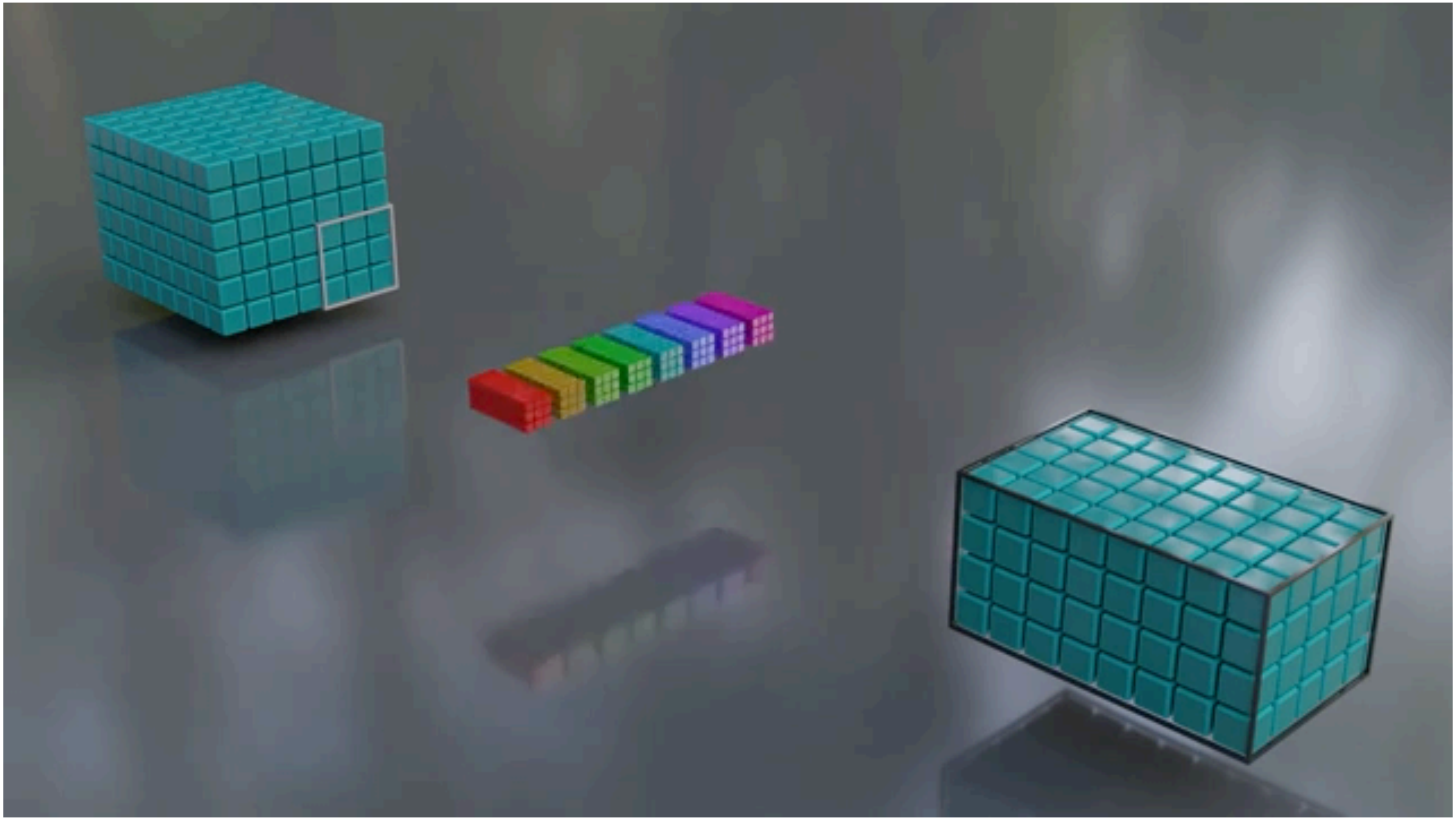
Today's Agenda

- Convolutional Neural Network (CNN)
  - Convolution operation
  - Nonlinearity
  - Pooling operation
  - CNN: convolutional layer + nonlinearity + pooling layer

How to calculate the output volume size?



How to calculate the output volume size?

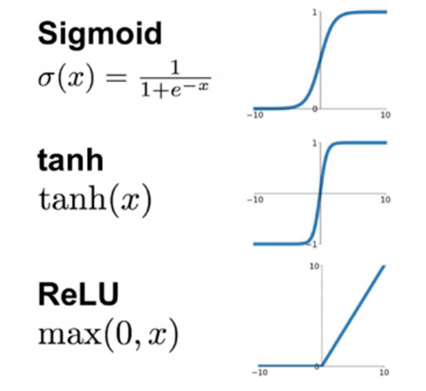


https://www.youtube.com/watch?v=w4kNHKcBGzA&t=210s

CNN: nonlinear activation function

- Convolutional Neural Network (CNN)
  - Convolution operation
  - Nonlinearity
  - Pooling operation
  - CNN: convolutional layer + nonlinearity + pooling layer

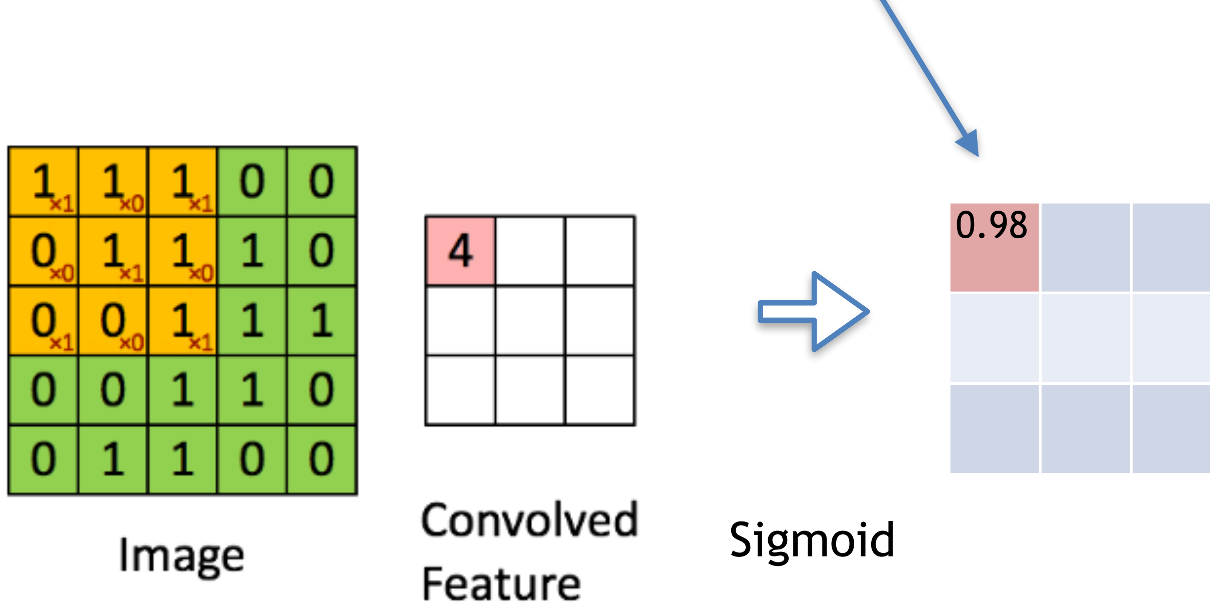
Nonlinear Function



- Just like an MLP, each convolutional output goes through a non-linear function such as Sigmoid, Tanh, or Rectified Linear Unit (ReLU)

$$convolution = 1*1+1*0+1*1+0*0+1*1+1*0+0*1+0*0+1*1 = 4$$

$$Sigmoid(4) = \frac{1}{1+exp(-4)} = 0.98$$



Today's Agenda

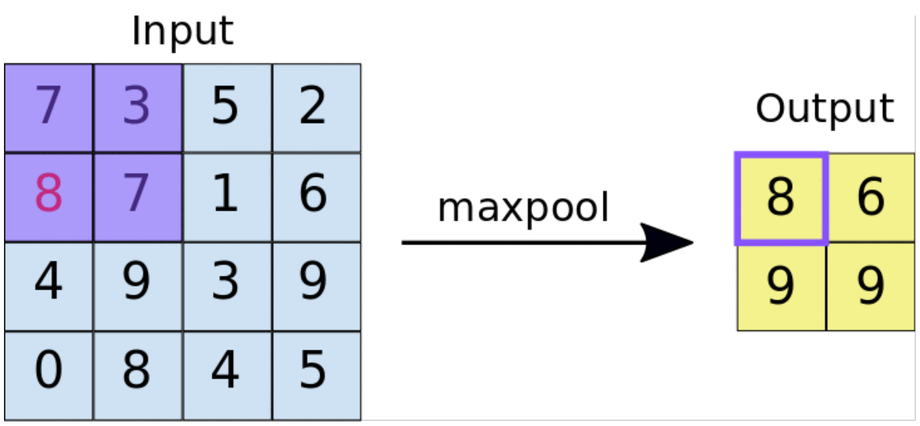
- Convolutional Neural Network (CNN)
  - Convolution operation
  - Nonlinearity
  - Pooling operation
  - CNN: convolutional layer + nonlinearity + pooling layer

Pooling Operation

- Image data can get really computationally inefficient, really quickly. To avoid this, we often toss in a layer that helps us to **summarize** and **downsample** the data

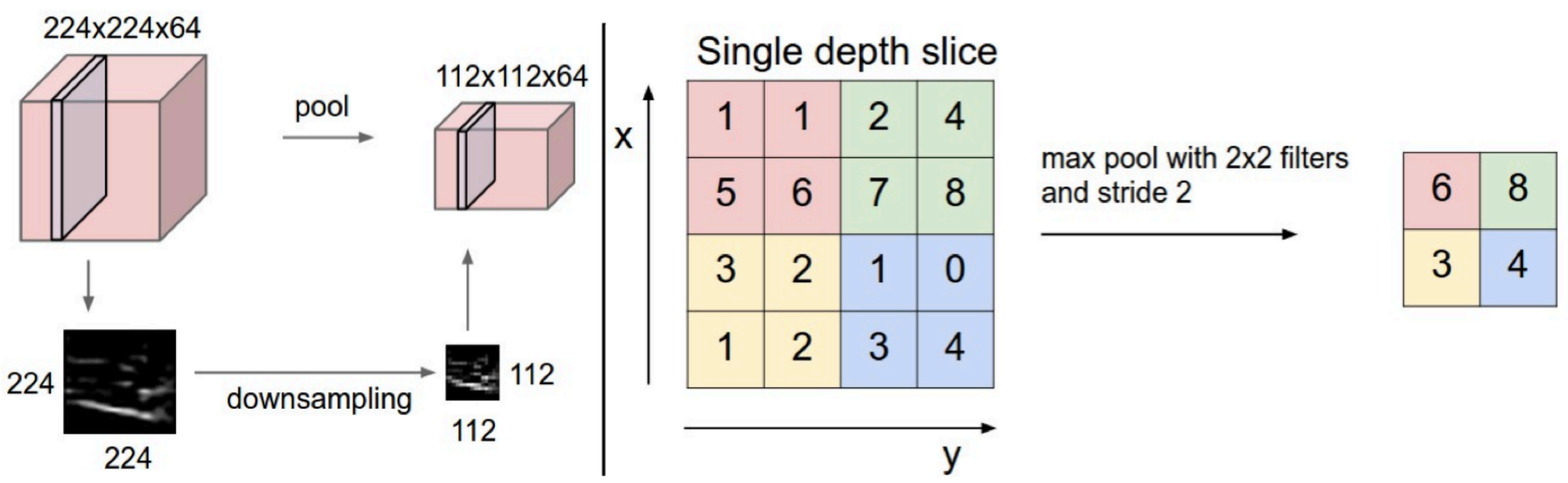
- In classical CNN, we find another useful operation called **pooling operation**

- A common pooling operation is **max pooling**, and its goal is to locally summarize the convolution. It performs something like a convolution, but rather than taking the dot product, it takes the maximum element in the filter area



Pooling Operation

- Pooling operation downsamples the volume spatially, **independently in each depth slice of the input volume**
- Besides max pooling, other pooling operations include: **sum pooling**, average pooling

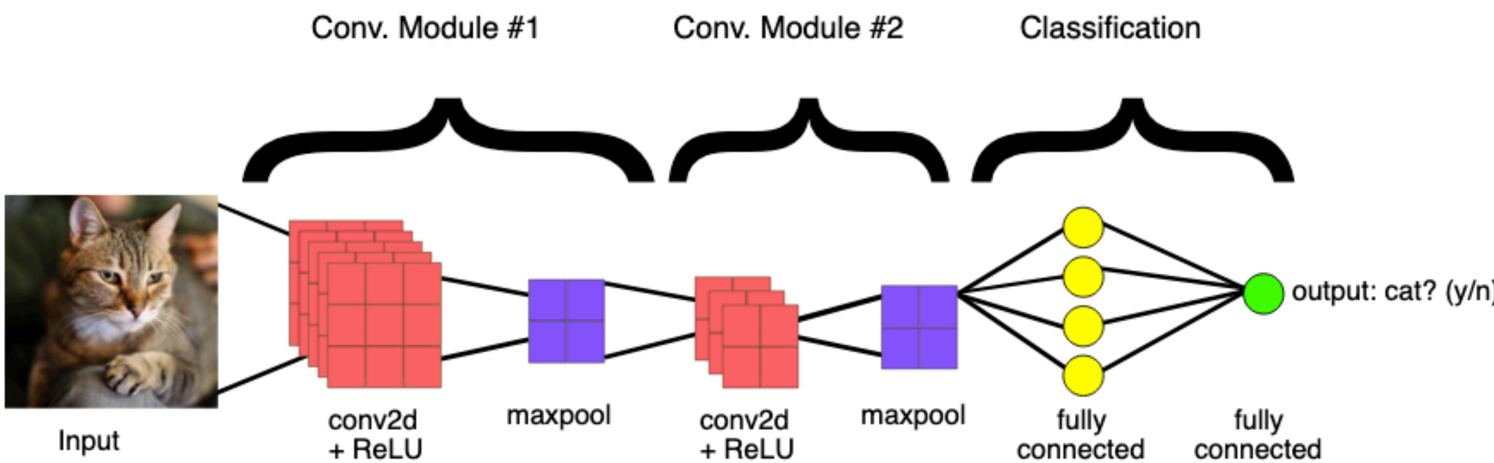


Today's Agenda

- Convolutional Neural Network (CNN)
  - Convolution operation
  - Nonlinearity
  - Pooling operation
  - CNN: convolutional layer + nonlinearity + pooling layer

CNN: A Composition of Convolutional Layers

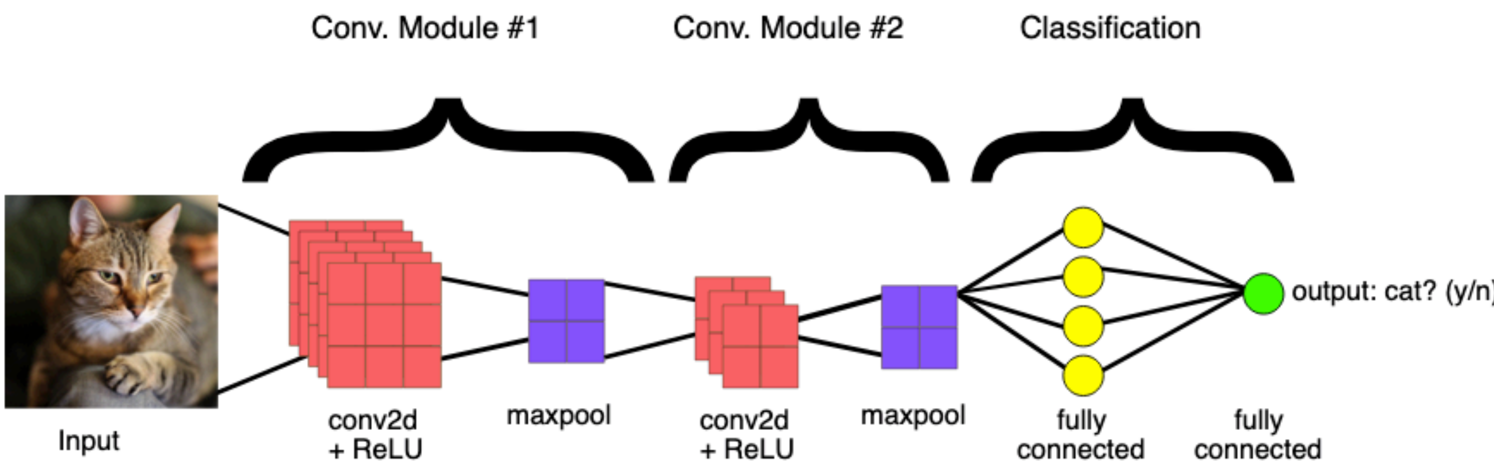
- We've talked about **image data, convolutions, nonlinearity, max pooling**, and how they are related to some computer vision tasks. Let's connect the dots
  - input is an image (in this case a color image, so 3 channels—red, green, and blue)
  - there are several filters, not just one.
  - `Conv2D` layers with `ReLU` are often followed by `maxpool`
  - towards the end of the model, we switch to fully connected (`Dense`) layer
  - We have as many output nodes as we have classes to predict



Reference

CNN: A Composition of Convolutional Layers

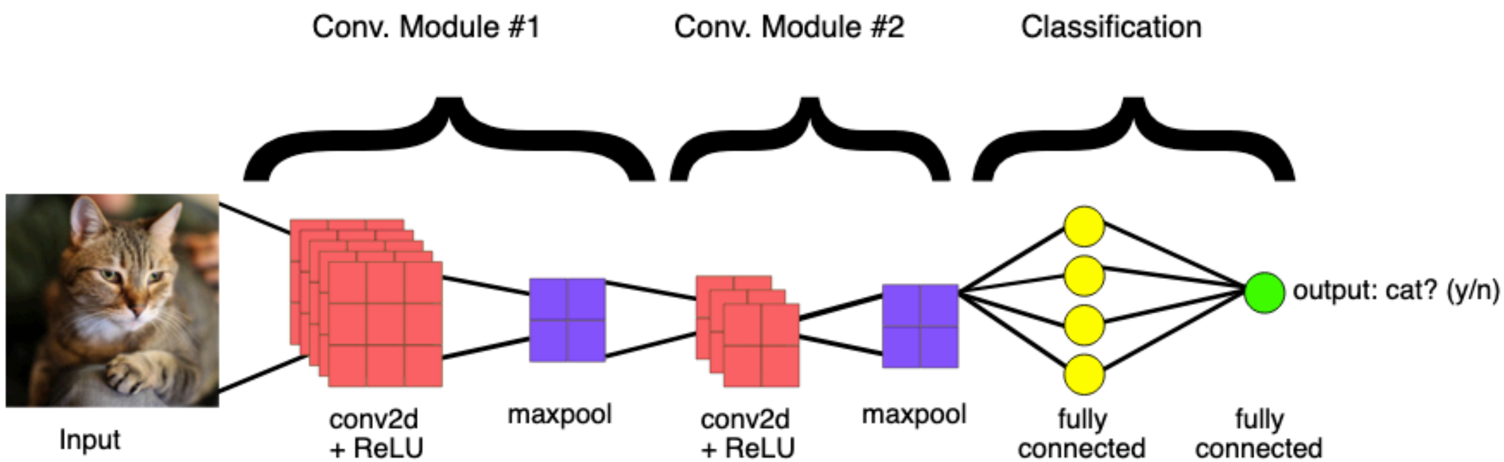
- **Big idea**: different kernels/filters can be used to extract specific information from the original image
- **Bigger idea**: Instead of using manually made kernels for feature extraction, through **deep CNNs we can learn these kernel values** (just like the weights of a traditional NN). These kernels can extract latent features
    - In MLP the way we learn is by changing the ***weights***
    - In CNNs, the way we learn is by changing the values in the **filters/kernels**



Reference

Convolutional Neural Network (CNN)

- Stack multiple stages of feature extractors
- Higher stages compute more global, more invariant features
- Linear layer (just like an MLP) at the end for the final classification

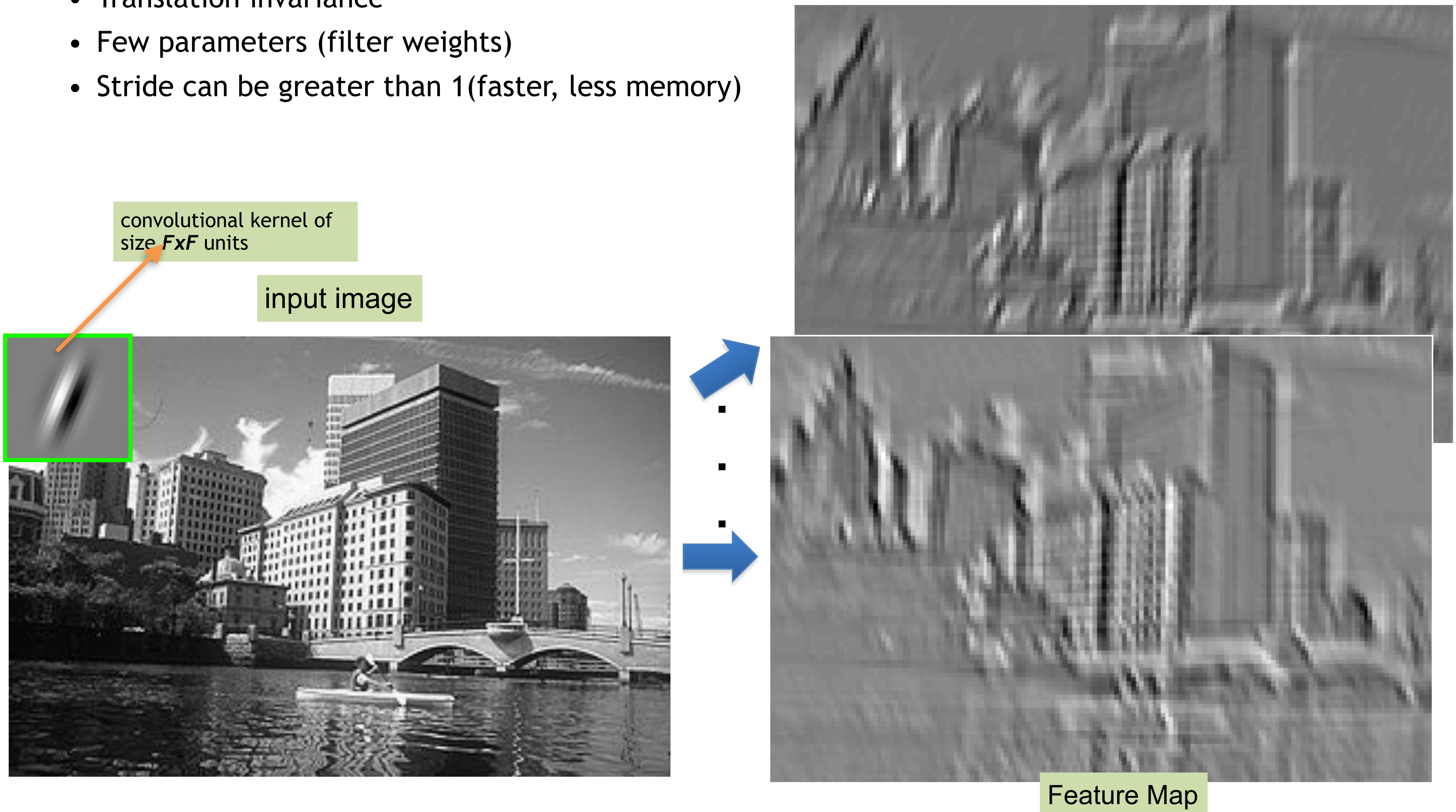


- Linear layer
- Fully connected layer
- Inner-product layer

Many names but they refer to the same thing

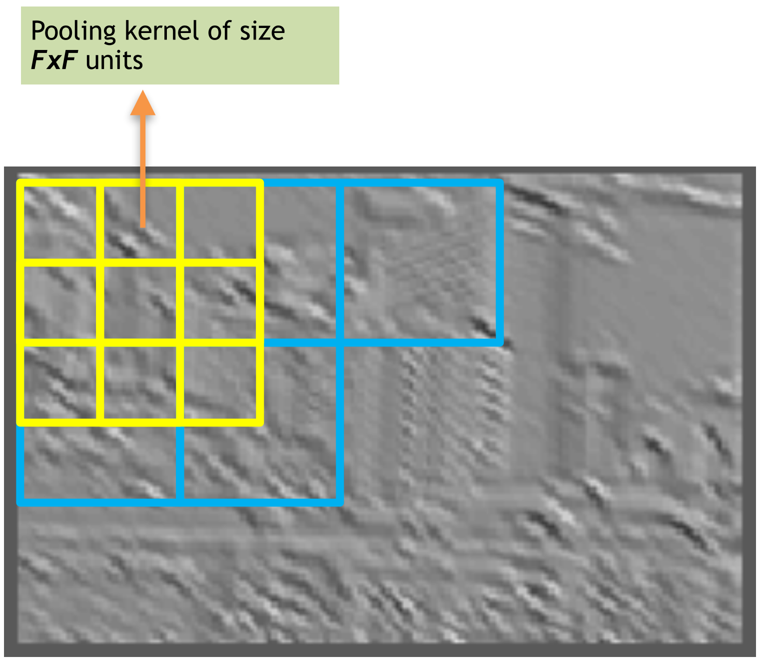
CNN: Applying Convolutional Filters

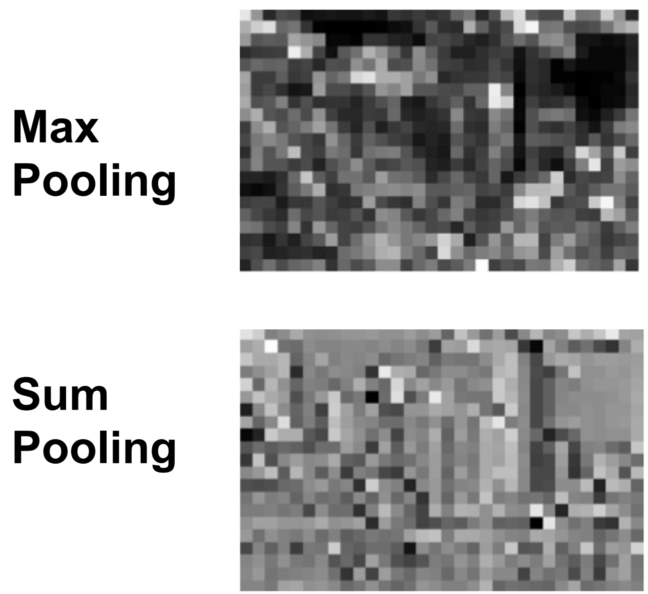
- Dependencies are local
- Translation invariance
- Few parameters (filter weights)
- Stride can be greater than 1(faster, less memory)



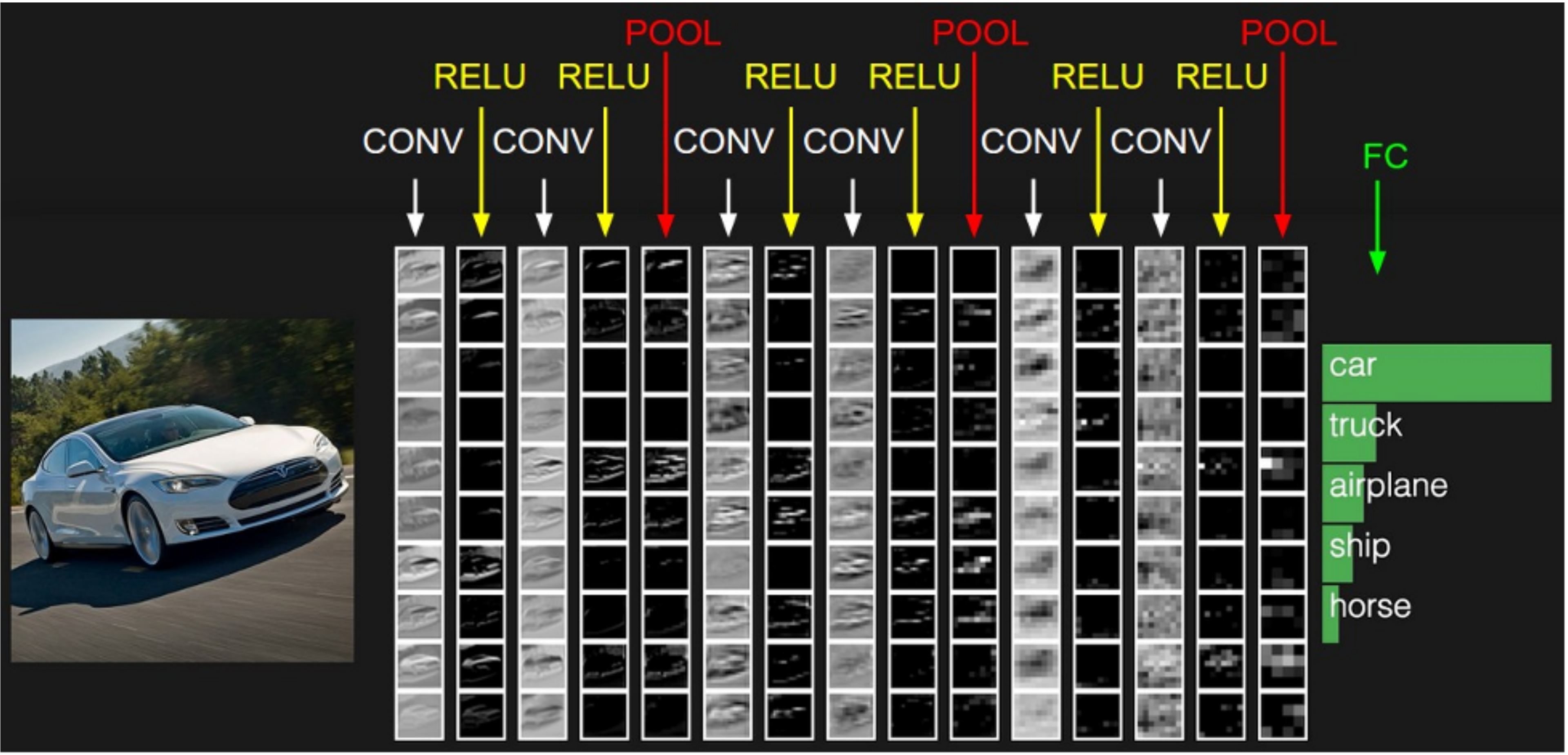
CNN: Applying Pooling Filters

- Sum pooling or max pooling
- Non-overlapping / overlapping regions
- Role of pooling:
  - Invariance to small transformations
  - Larger receptive fields (see more of input)



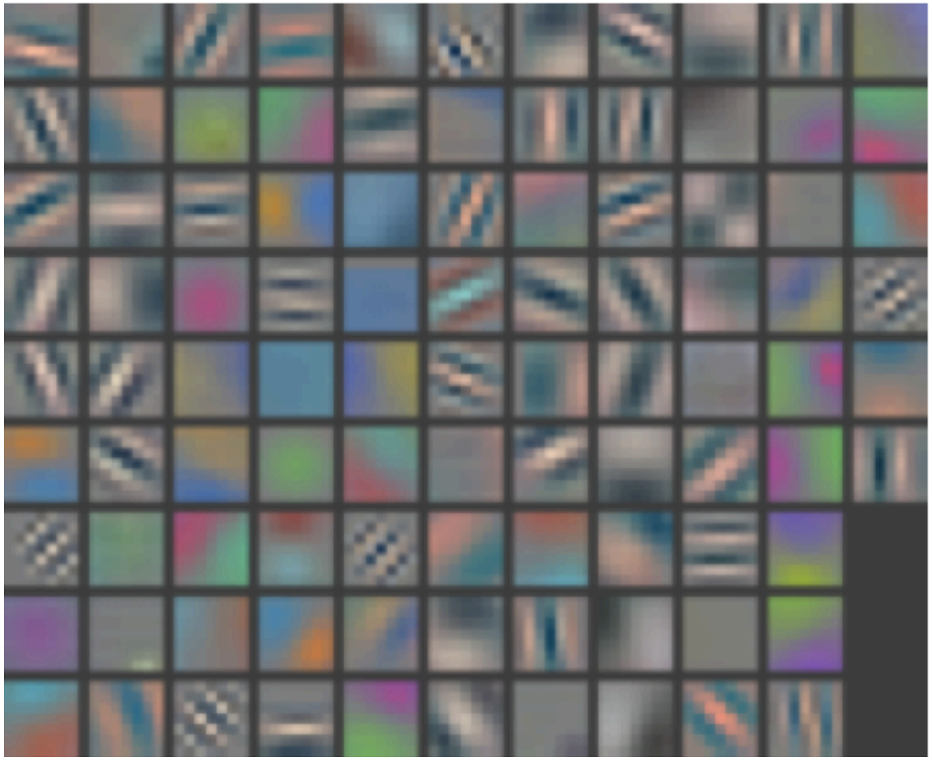


CNN: A Composition of Convolutional Layers



What does each layer learn?

Layer 1 Filters



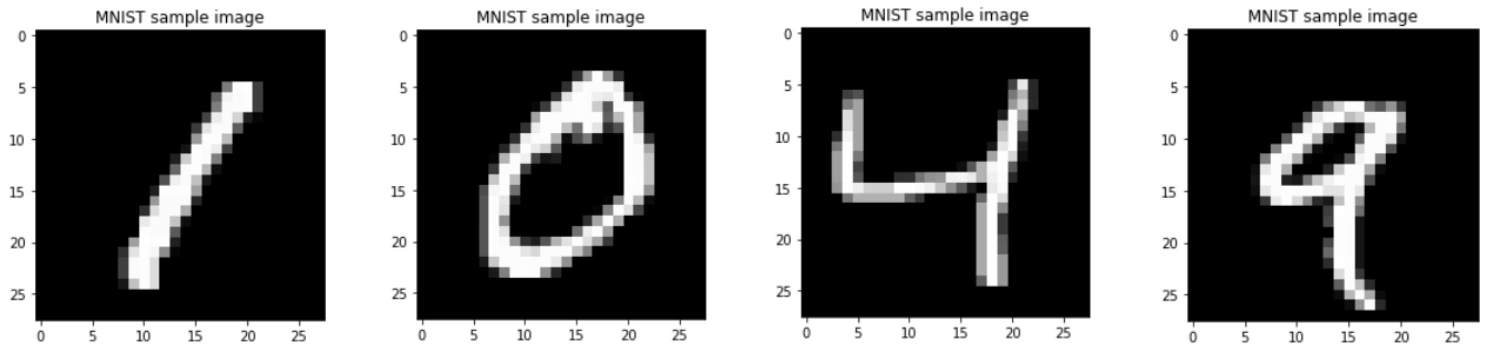
Notice that the parameter sharing assumption is relatively reasonable: If detecting a horizontal edge is important at some location in the image, it should intuitively be useful at some other location as well due to the translationally-invariant structure of images. There is therefore no need to relearn to detect a horizontal edge at every one of the 55*55 distinct locations in the Conv layer output volume.

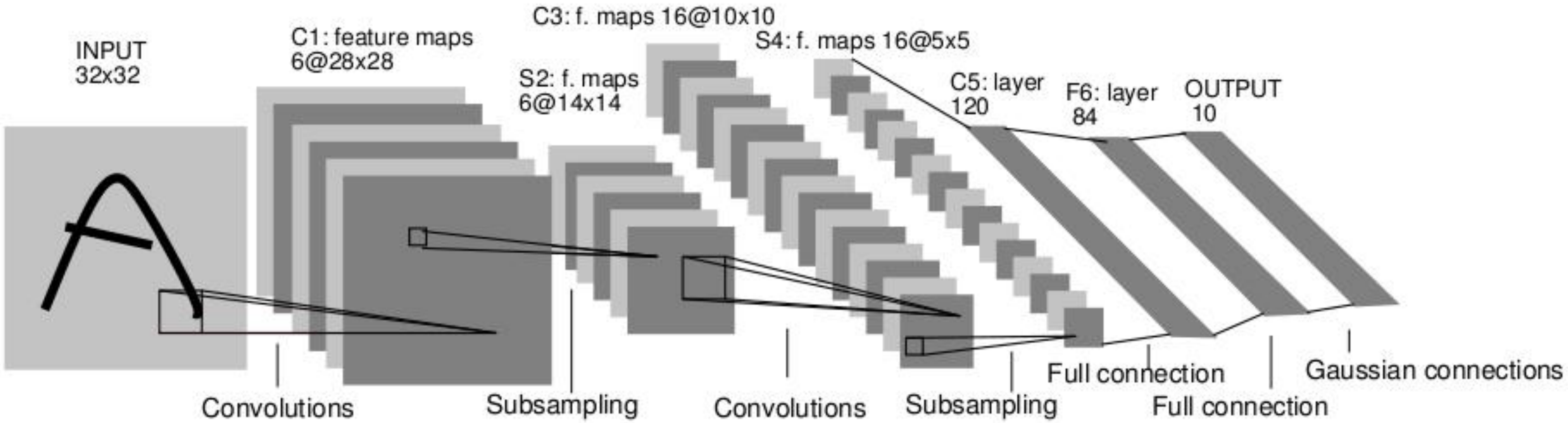
Today's Agenda

- Convolutional Neural Network (CNN)
    - Convolution operation
    - Nonlinearity
    - Pooling operation
    - CNN: convolutional layer + nonlinearity + pooling layer
- Popular CNNs
    - LeNet
    - AlexNet
    - VGG
    - ResNet

Popular CNN: LeNet

- LeNet is a simple CNN architecture suitable for well-structured image
  - e.g., 28x28 pixels image of digits from 0 to 9 in MNIST or our Fashion-MNIST dataset

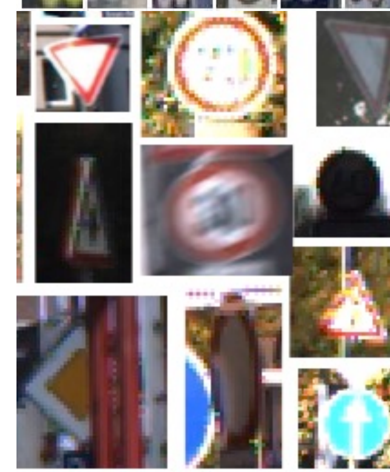




CNNs Success

- Handwritten text/digits
  - MNIST (0.17% error [Ciresan et al. 2011]
  - Arabic & Chinese [Ciresan et al. 2012]

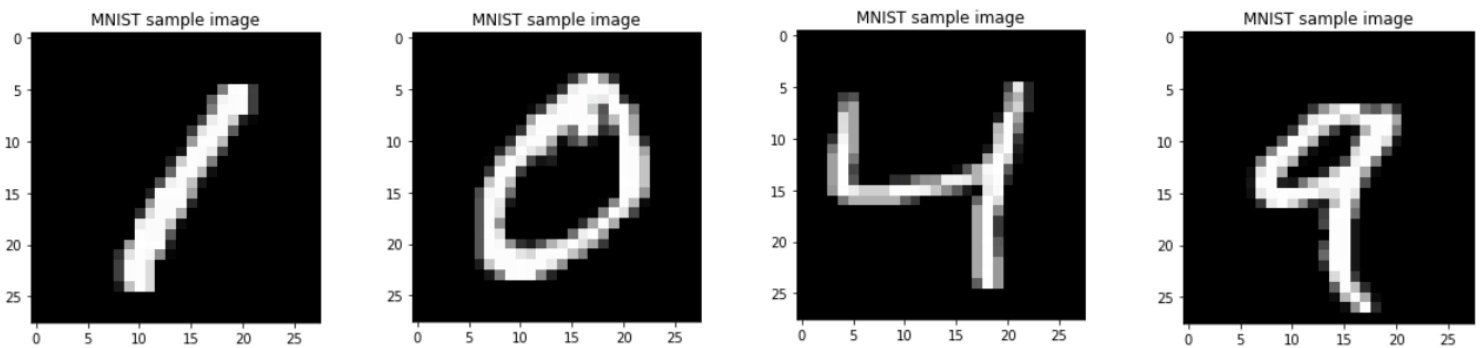
- Simpler recognition benchmarks
  - CIFAR-10 (9.3% error [Wan et al. 2013])
  - Traffic sign recognition



- Until 2011, it was less good at more complex datasets
  - Caltech-101/256 (few training examples)

Popular CNN: LeNet

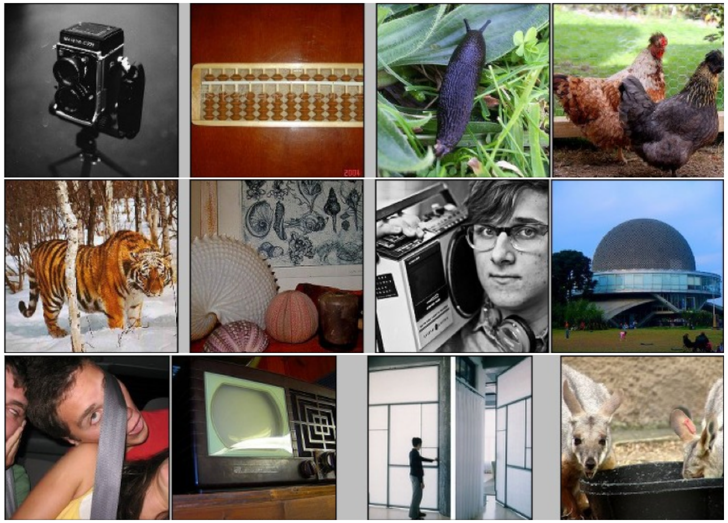
- LeNet is a simple CNN architecture suitable for well-structured image
    - e.g., 28x28 pixels image of digits from 0 to 9 in MNIST or our Fashion-MNIST dataset



- Real-world images are much more complicated; pose challenges in classification
    - e.g., high resolution images 600x480 pixels image and contents have a lot more diversity

ImageNet Challenge 2012

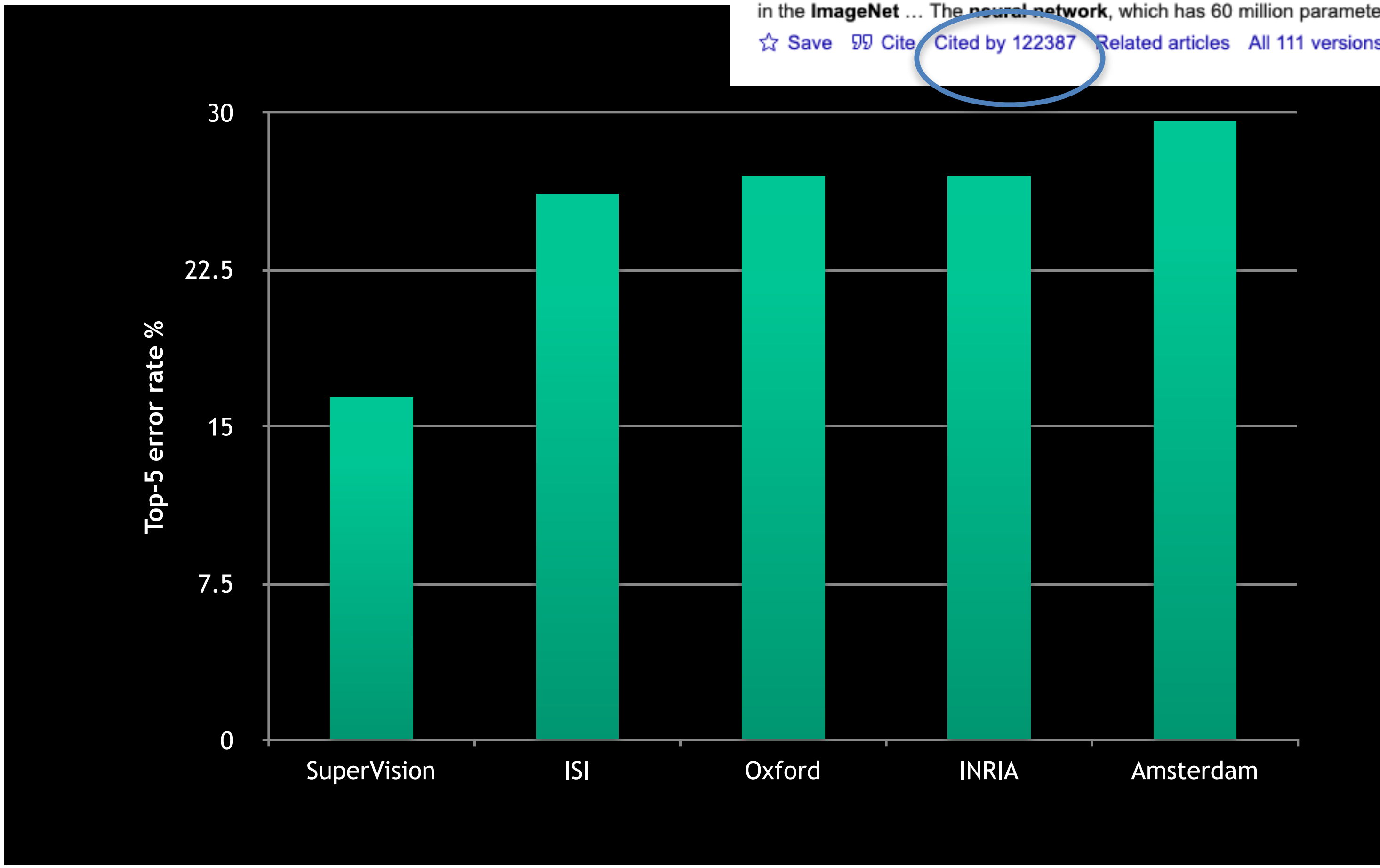
- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon Turk
- **Challenge: 1.2 million training images, 1000 classes**



[Deng et al. CVPR 2009]

ImageNet Challenge 2012

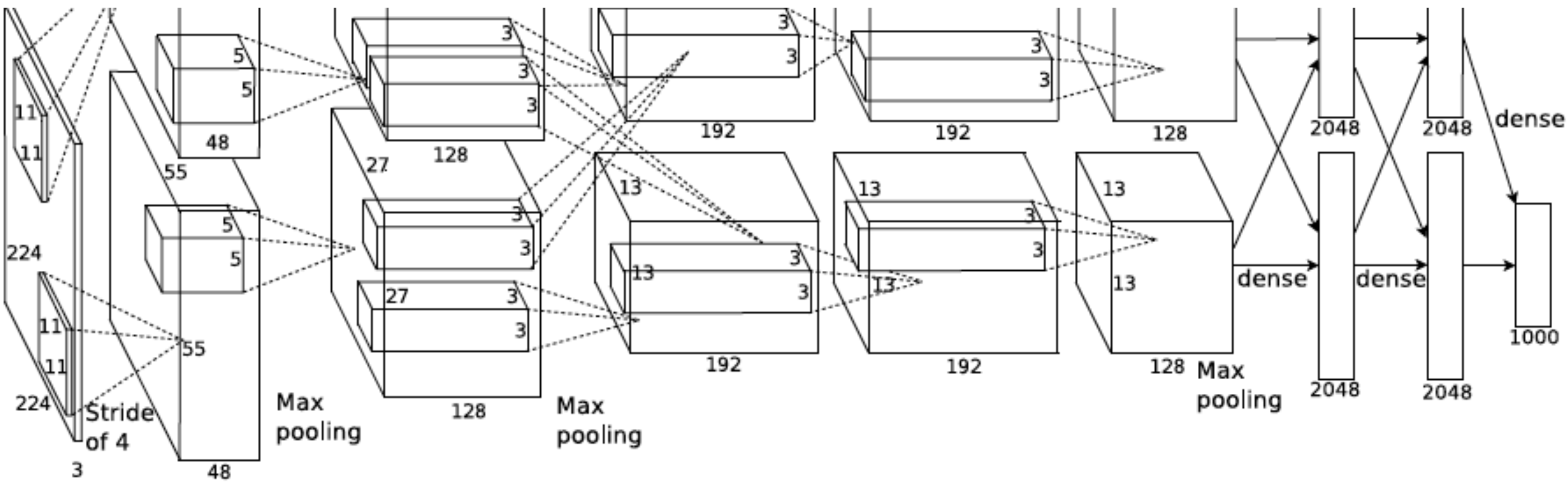
Imagenet classification with deep convolutional neural networks
A Krizhevsky, I Sutskever… - Advances in **neural** …, 2012 - proceedings.neurips.cc
… a large, **deep convolutional neural network** to **classify** the 1.2 million high-resolution images in the **ImageNet** … The **neural network**, which has 60 million parameters and 650,000 neurons, …
Save Cite Cited by 122387 Related articles All 111 versions



- AlexNet (Krizhevsky et al.) -- **16.4% error** (top-5)
- Next best (non-convnet) – **26.2% error**

Popular CNN: AlexNet

- Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week
  - Better regularization for training (DropOut)



A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Popular CNN: AlexNet

Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2

} self.features

[6x6x256] **ADAPTIVE AVG POOL**: filters with output size 6x6

} self.avgpool

[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

} self.classifier

A. Krizhevsky, I. Sutskever, and G. Hinton, [ImageNet Classification with Deep Convolutional Neural Networks](), NIPS 2012

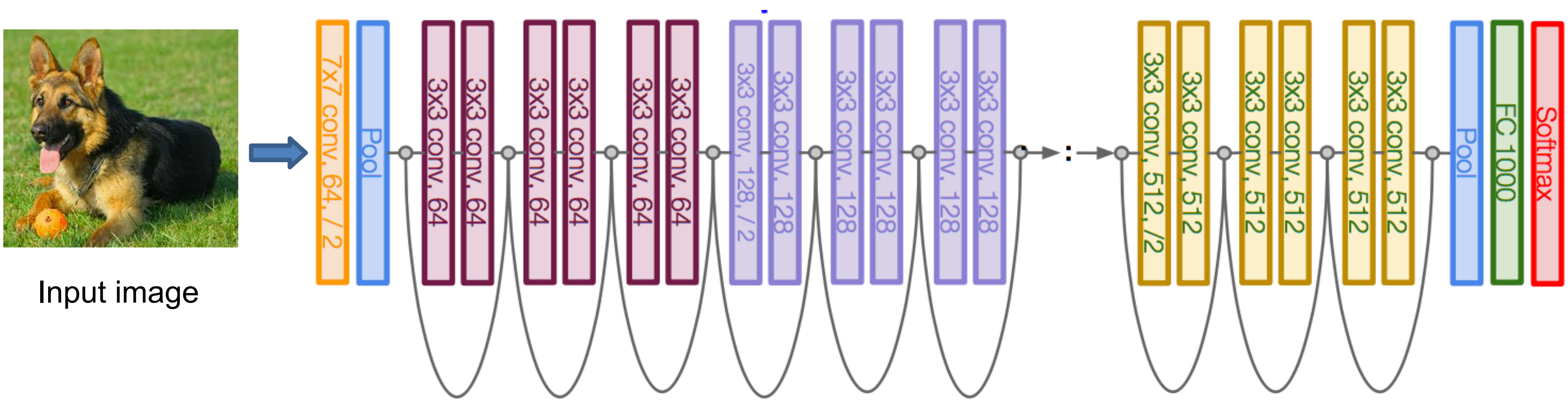
Popular CNN: VGG

- VGG was the winner of ImageNet (1000-class image classification) challenge in 2014
  - proposed by Andrew Zisserman's group in Oxford University

Very Deep Convolutional Networks for Large-Scale Image Recognition - Karen Simonyan and Andrew Zisserman

Popular CNN: ResNet

- ResNet was the winner of ImageNet challenge in 2015



[**Deep Residual Learning for Image Recognition**]() **- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun**

ImageNet Winners by the Popular CNNs

- AlexNet (2012) —> VGG (2014)—> ResNet (2015)