Lecture 7: Convolutional Neural Networks

Pranabendu Misra
Chennai Mathematical Institute

Advanced Machine Learning 2022

(based on slides by Madhavan Mukund)

Deep Neural Networks for Recognizing Images

Last Lecture:

- We trained a Fully-Connected DNN to recognize handwritten digits from the MNIST Dataset
- It performed fairly well (97.5% Test Accuracy)
- Similar networks, but perhaps with larger number of neurons, can be built for more complex image classification tasks.
- However, we also saw that the Fully-Connected Network doesn't use *Visual Information*.
  - We fixed an arbitrary permutation, and scrambled all training and test images using it. The resulting images were no longer recognizable as digits (by us humans).
  - However the Fully-Connected network still managed a 97.5% Test accuracy. This means this network was not using visual information.
- Can we do better by using some visual information in the images?
- How can this be done?

How the brain recognizes images

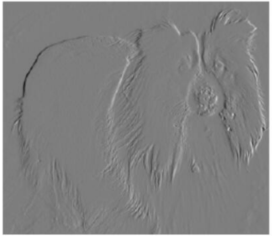
- Visual cortex processes images
- Experiments on cats and monkeys [Hubel, Wiesel 1959], Nobel Prize 1981
- Visual cortex organized in layers

How the brain recognizes images

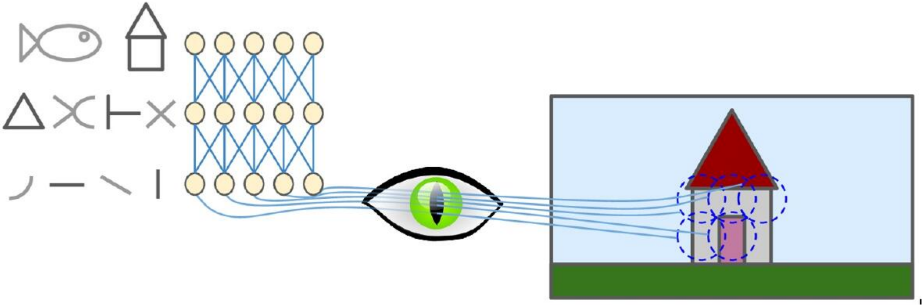
- Visual cortex processes images
- Experiments on cats and monkeys [Hubel, Wiesel 1959], Nobel Prize 1981
- Visual cortex organized in layers
  - Each layer detects features
  - Initial layers detect simple features — edges
  - Later layers combine features of earlier layers — detect contours, shapes, entire object

How the brain recognizes images

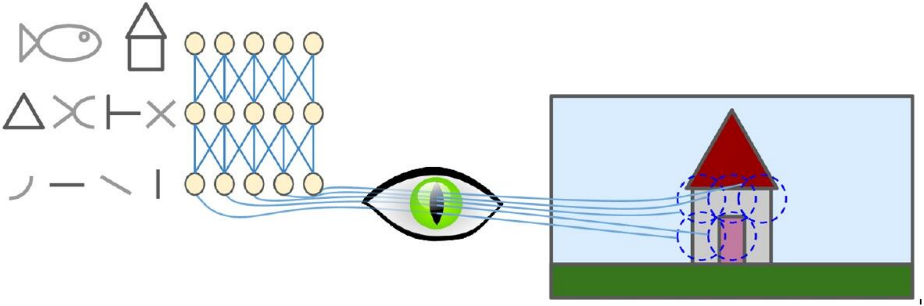
- Visual cortex processes images
- Experiments on cats and monkeys [Hubel, Wiesel 1959], Nobel Prize 1981
- Visual cortex organized in layers
    - Each layer detects features
    - Initial layers detect simple features — edges
    - Later layers combine features of earlier layers — detect contours, shapes, entire object
- Convolutional neural network (CNN) — layered network



Receptive field

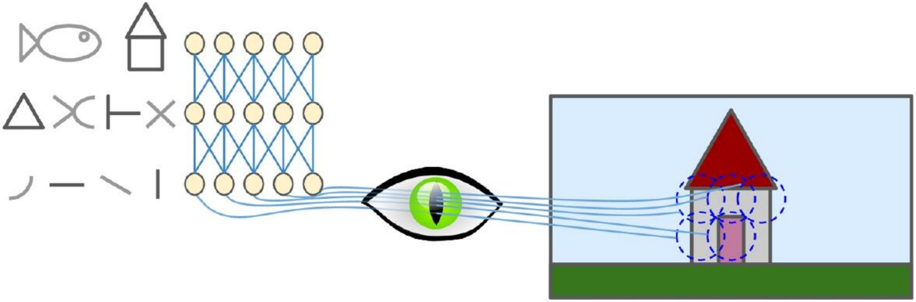


- Each neuron focuses on a small region — receptive field



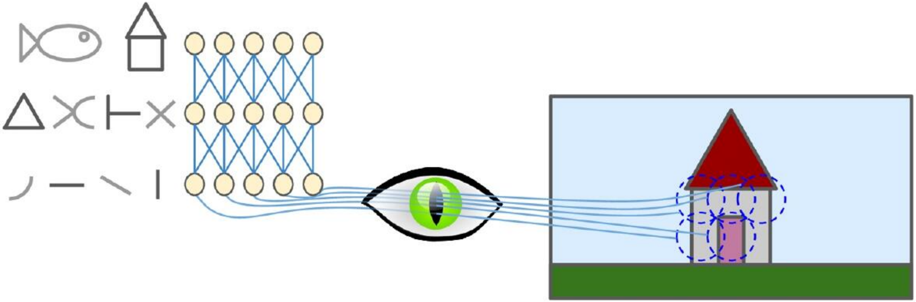
- Each neuron focuses on a small region — receptive field
- Vanilla neural network reads entire image as input
  - MNIST — 28×28 pixels

Receptive field



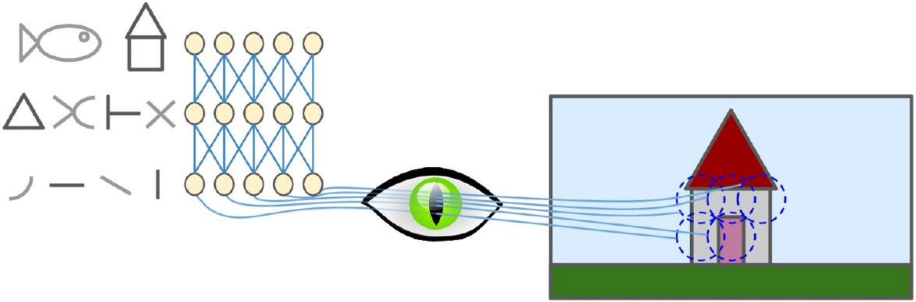
- Each neuron focuses on a small region — receptive field
- Vanilla neural network reads entire image as input
  - MNIST — 28×28 pixels
- Colour image, 200×200

Receptive field



- Each neuron focuses on a small region — receptive field
- Vanilla neural network reads entire image as input
  - MNIST — 28×28 pixels
- Colour image, 200×200
  - Three colours — $200 \times 200 \times 3$ inputs
  - Each neuron in first layer has $120,000$ input weights
  - Multiple such neurons

Receptive field



- Each neuron focuses on a small region — receptive field
- Vanilla neural network reads entire image as input
  - MNIST — 28×28 pixels
- Colour image, 200×200
  - Three colours — $200 \times 200 \times 3$ inputs
  - Each neuron in first layer has $120,000$ input weights
  - Multiple such neurons
- Parameter blowup, overfitting

Filters and convolution

- Aggregate values over a region
  - Smoothening — take average
  - Vertical lines — difference between adjacent columns
  - Horizontal lines — difference between adjacent rows
- Pass a filter f over the image
  - Convolution — $I * f$
  - Sometimes, filter is called a convolution kernel — $I * K$

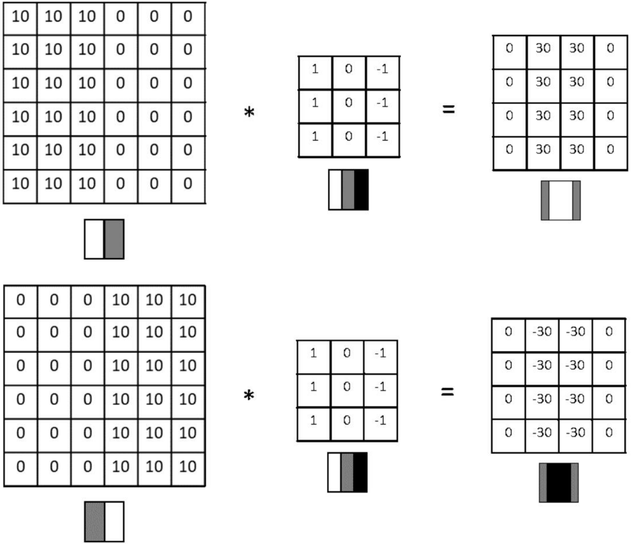
Filters and convolution

- Aggregate values over a region
  - Smoothening — take average
  - Vertical lines — difference between adjacent columns
  - Horizontal lines — difference between adjacent rows
- Pass a filter f over the image
  - Convolution — $I * f$
  - Sometimes, filter is called a convolution kernel — $I * K$
- Light to dark vertical edges

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$

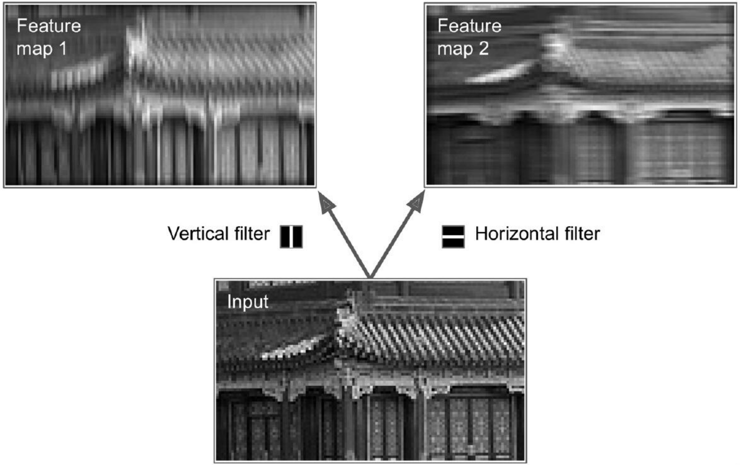
Filters and convolution

- Aggregate values over a region
  - Smoothening — take average
  - Vertical lines — difference between adjacent columns
  - Horizontal lines — difference between adjacent rows
- Pass a filter f over the image
  - Convolution — $I * f$
  - Sometimes, filter is called a convolution kernel — $I * K$
- Light to dark vertical edges
- Dark to light vertical edges



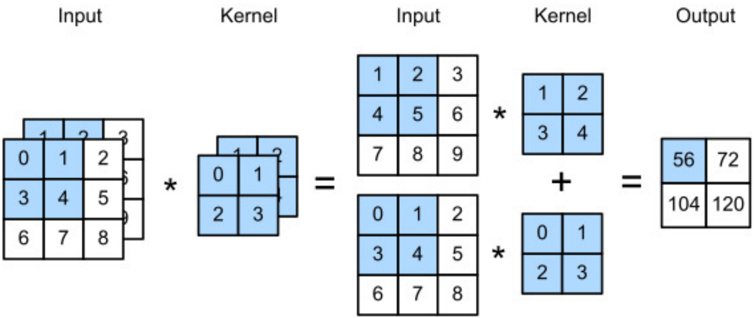
Feature maps

- Filters produce feature maps



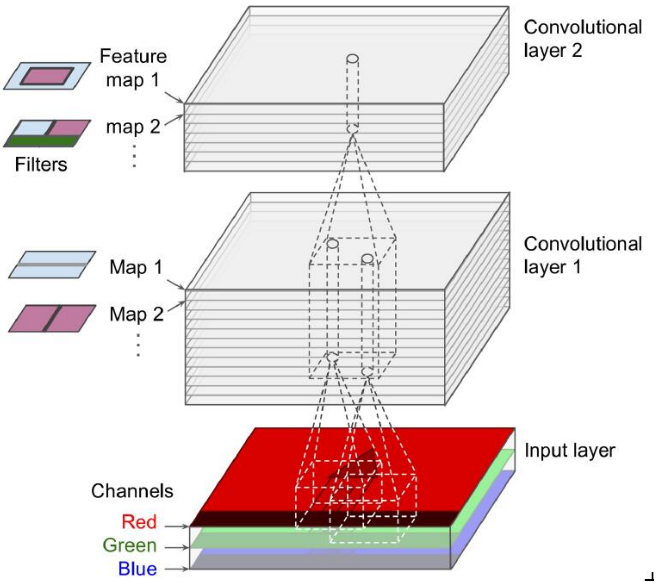
Multi-Channel Inputs

- Real World Images are in color (3-channels)
- Similarly convolution filters in higher layer will need to work with feature maps produced by multiple lower layer convolutions.
- So we need Multi-Channel Convolution



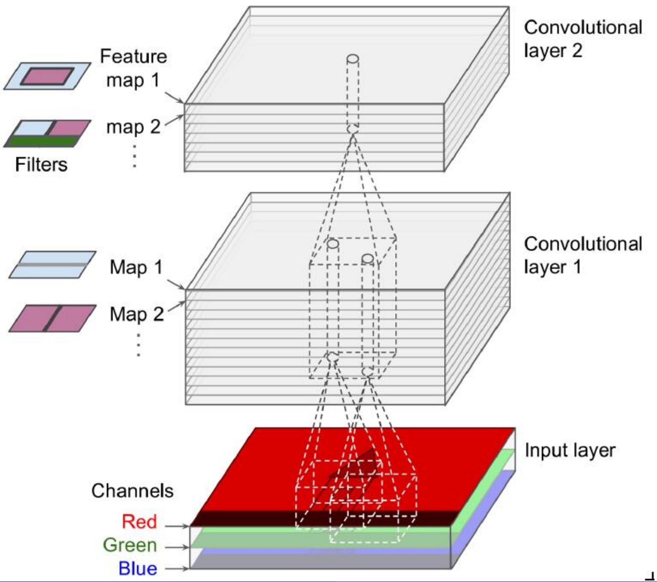
Feature maps

- Filters produce feature maps
- Colour images are split by channel



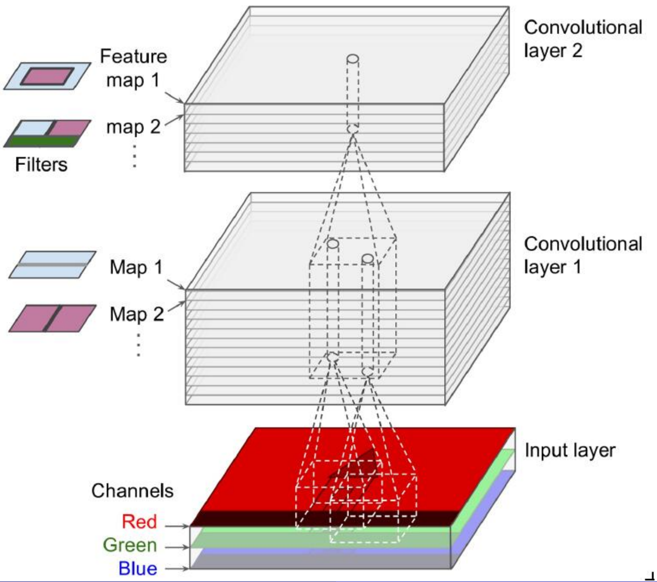
Feature maps

- Filters produce feature maps
- Colour images are split by channel
- Each layer has many feature maps
  - Array of filters, each connected to a different region



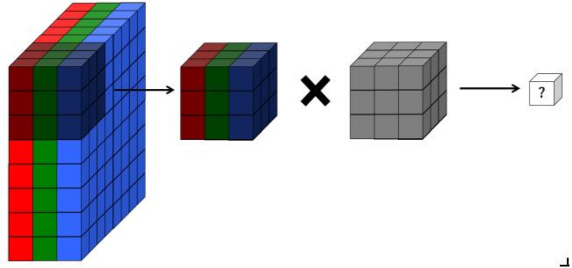
Feature maps

- Filters produce feature maps
- Colour images are split by channel
- Each layer has many feature maps
  - Array of filters, each connected to a different region
- Higher layers combine features discovered by lower layers



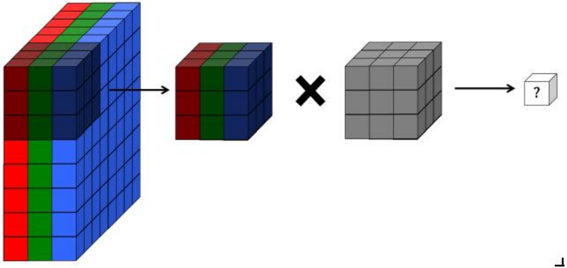
Volumetric view

- Each filter processes a volume of inputs



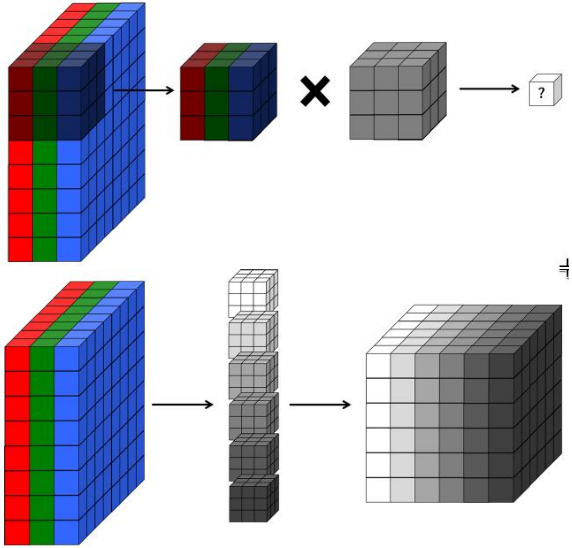
Volumetric view

- Each filter processes a volume of inputs
- Each layer has sublayers
  - A sublayer is an array of such filters



Volumetric view

- Each filter processes a volume of inputs
- Each layer has sublayers
    - A sublayer is an array of such filters
- Each layer produces a block of outputs

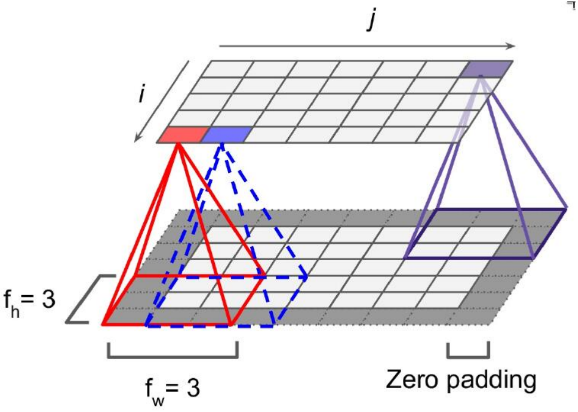


Zero padding, stride

- Each filter f has height f_h, width f_w
    - Receptive field of f

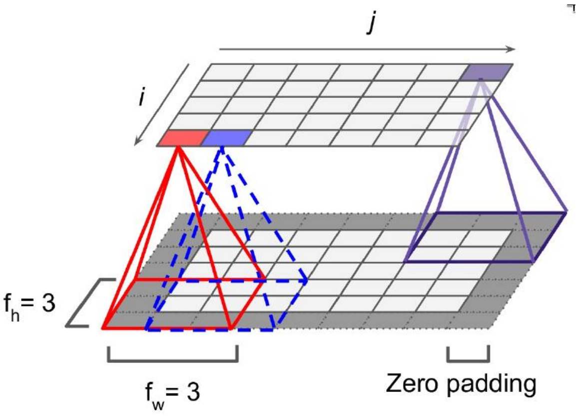
Zero padding, stride

- Each filter f has height f_h, width f_w
  - Receptive field of f
- Need to extend the boundary for filter to work properly at the edges
  - Zero padding



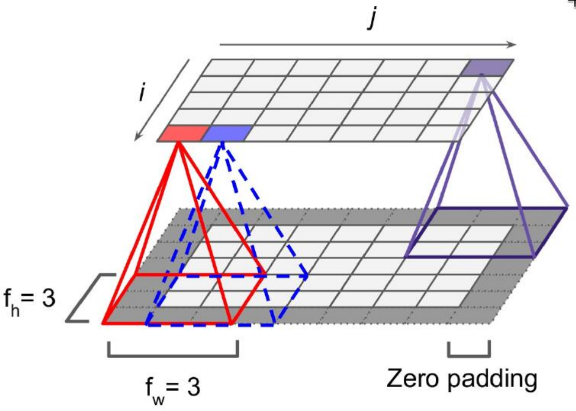
Zero padding, stride

- Each filter f has height f_h, width f_w
  - Receptive field of f
- Need to extend the boundary for filter to work properly at the edges
  - Zero padding
- With padding, feature map has same dimension as input



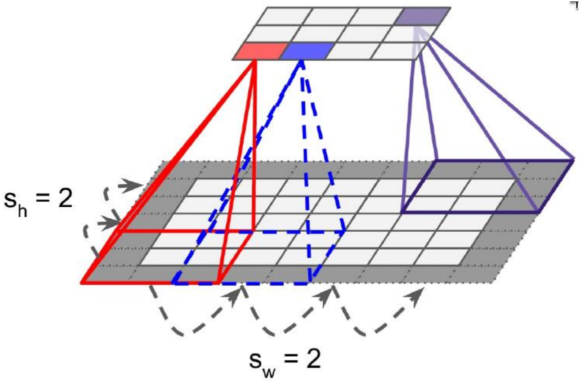
Zero padding, stride

- Each filter f has height f_h, width f_w
  - Receptive field of f
- Need to extend the boundary for filter to work properly at the edges
  - Zero padding
- With padding, feature map has same dimension as input
- Note: In an actual CNN, filters are not designed by hand
  - Fix f_h and f_w, but weights are learned from training data



Zero padding, stride

- Each filter f has height f_h, width f_w
  - Receptive field of f
- Need to extend the boundary for filter to work properly at the edges
  - Zero padding
- With padding, feature map has same dimension as input
- To reduce dimension, we can space out the receptive fields
  - Horizontal and vertical stride

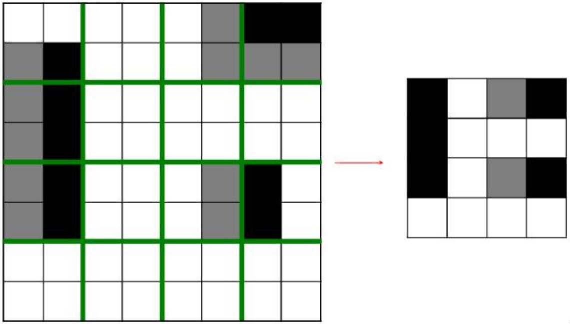


Pooling

- Filters process overlapping regions
- Pooling processes partitions
  - Subsampling, reduce dimensionality

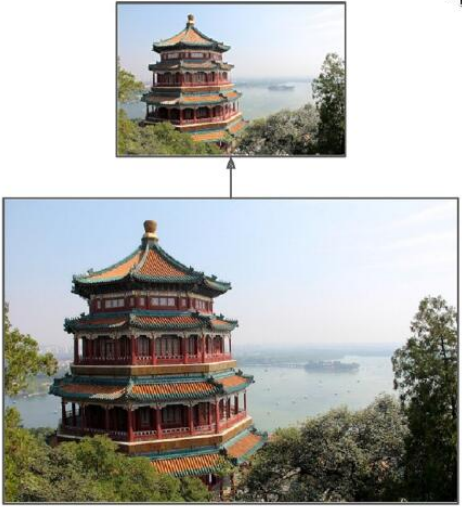
Pooling

- Filters process overlapping regions
- Pooling processes partitions
  - Subsampling, reduce dimensionality
- Most common is max-pool over 2×2 window



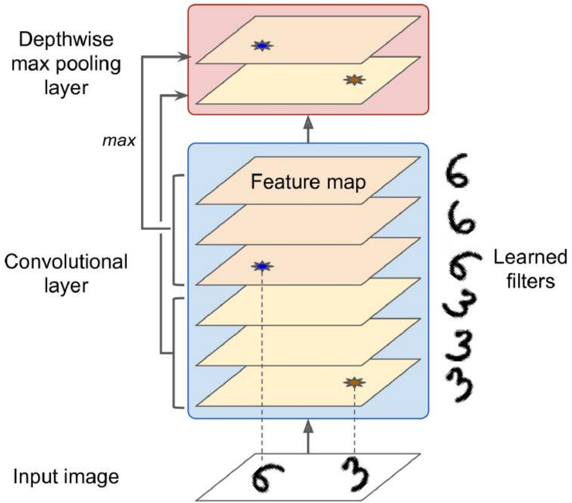
Pooling

- Filters process overlapping regions
- Pooling processes partitions
  - Subsampling, reduce dimensionality
- Most common is max-pool over 2×2 window
- Here, max-pooling reduces an image to half its size



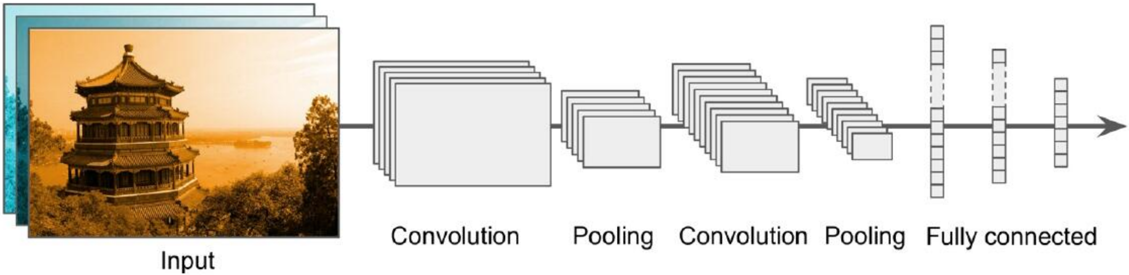
Pooling

- Filters process overlapping regions
- Pooling processes partitions
  - Subsampling, reduce dimensionality
- Most common is max-pool over 2×2 window
- Here, max-pooling reduces an image to half its size
- Can also pool depthwise — for instance, to learn features invariant to rotation



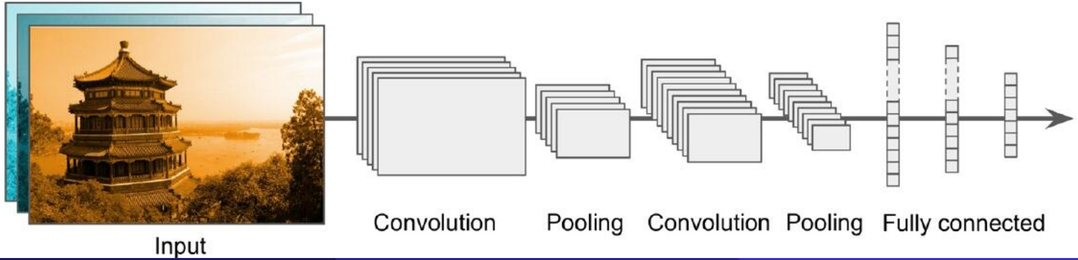
Typical CNN Architecture

- A typical CNN has multiple iterations of convolution followed by pooling
- After final pooling, conventional completely connected network



Parameter sharing

- A filter is a layer of identical nodes operating on different regions (receptive fields)
- All these nodes should behave the same
- While training, their weights are tied to each other — parameter sharing
- Thus, backward pass of backpropagation calculation is reduced
- Forward pass needs to compute individual outputs — still expensive



CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service

CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service
- AlexNet — Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012
  - ImageNet, 14 million images, 20,000 categories, hand annotated
  - Top-five error rate — at least one of top 5 prediced labels is correct

CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service
- AlexNet — Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012
  - ImageNet, 14 million images, 20,000 categories, hand annotated
  - Top-five error rate — at least one of top 5 prediced labels is correct
  - 2012 ImageNet challenge, AlexNet reduced top-five error rate from 26% to 17%

CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service
- AlexNet — Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012
  - ImageNet, 14 million images, 20,000 categories, hand annotated
  - Top-five error rate — at least one of top 5 prediced labels is correct
  - 2012 ImageNet challenge, AlexNet reduced top-five error rate from 26% to 17%
  - First to add multiple convolution layers between pooling layers
  - Also some normalization layers

CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service
- AlexNet — Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012
  - ImageNet, 14 million images, 20,000 categories, hand annotated
  - Top-five error rate — at least one of top 5 prediced labels is correct
  - 2012 ImageNet challenge, AlexNet reduced top-five error rate from 26% to 17%
  - First to add multiple convolution layers between pooling layers
  - Also some normalization layers
- GoogLeNet — Christian Szegedy et al, 2014
  - 2014 ImageNet challenge, reduced top-five error rate to 7%

CNNs through the ages

- LeNet-5 — Yann LeCun, 1998
  - Handwritten digits, MNIST data set from US postal service
- AlexNet — Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, 2012
  - ImageNet, 14 million images, 20,000 categories, hand annotated
  - Top-five error rate — at least one of top 5 prediced labels is correct
  - 2012 ImageNet challenge, AlexNet reduced top-five error rate from 26% to 17%
  - First to add multiple convolution layers between pooling layers
  - Also some normalization layers
- GoogLeNet — Christian Szegedy et al, 2014
  - 2014 ImageNet challenge, reduced top-five error rate to 7%
  - Inception layer with 1×1 filters, operates in depth dimension, cross-channel features

CNNs through the ages

- ResNet — Kaiming He et al, 2015
  - 2014 ImageNet challenge, reduced top-five error rate to under 3.6%
  - 152 layers!

CNNs through the ages

- ResNet — Kaiming He et al, 2015
  - 2014 ImageNet challenge, reduced top-five error rate to under 3.6%
  - 152 layers!
  - Skip connections to speed up learning
    - Input to a layer is added to output of a higher layer
    - Higher layer learns $h(x) - x$ rather than $h(x)$ — residual learning
    - Accelerates learning through multiple layers

CNNs through the ages

- ResNet — Kaiming He et al, 2015
  - 2014 ImageNet challenge, reduced top-five error rate to under 3.6%
  - 152 layers!
  - Skip connections to speed up learning
    - Input to a layer is added to output of a higher layer
    - Higher layer learns $h(x) - x$ rather than $h(x)$ — residual learning
    - Accelerates learning through multiple layers
- Xception, Chollet, 2016
- SENet, Hu et al, 2017

Summary

- CNN architecture mimics the visual cortex
  - Processing is done in layers
  - Filters aggregate information across a small receptive field to capture features
  - Higher layers combine features at lower levels

Summary

- CNN architecture mimics the visual cortex
  - Processing is done in layers
  - Filters aggregate information across a small receptive field to capture features
  - Higher layers combine features at lower levels
- Pooling layers use subsampling for dimension reduction

Summary

- CNN architecture mimics the visual cortex
  - Processing is done in layers
  - Filters aggregate information across a small receptive field to capture features
  - Higher layers combine features at lower levels
- Pooling layers use subsampling for dimension reduction
- Within a filter layer, parameter sharing reduces number of parameters to be learned

Summary

- CNN architecture mimics the visual cortex
  - Processing is done in layers
  - Filters aggregate information across a small receptive field to capture features
  - Higher layers combine features at lower levels
- Pooling layers use subsampling for dimension reduction
- Within a filter layer, parameter sharing reduces number of parameters to be learned
- AlexNet signalled resurgence of CNNs with huge margin of victory in ImageNet 2012

Summary

- CNN architecture mimics the visual cortex
  - Processing is done in layers
  - Filters aggregate information across a small receptive field to capture features
  - Higher layers combine features at lower levels
- Pooling layers use subsampling for dimension reduction
- Within a filter layer, parameter sharing reduces number of parameters to be learned
- AlexNet signalled resurgence of CNNs with huge margin of victory in ImageNet 2012
- CNNs continue to evolve with specialized hacks