Mathematical techniques in data science

Lecture 12: Convolutional neural networks

Reminders

- Office hours:
  - MW 3pm-4pm, Ewing Hall 312
  - By appointments
- Homework 1: due Monday EOD
- Sign up for group projects before class meeting next week

Feed-forward neural networks

- Structure:
  - Graphical representation
  - Activation functions
  - Loss functions
- Training:
  - Stochastic gradient descent
  - Back-propagation

- High level API for deep learning
- More flexible to define network architecture than sklearn

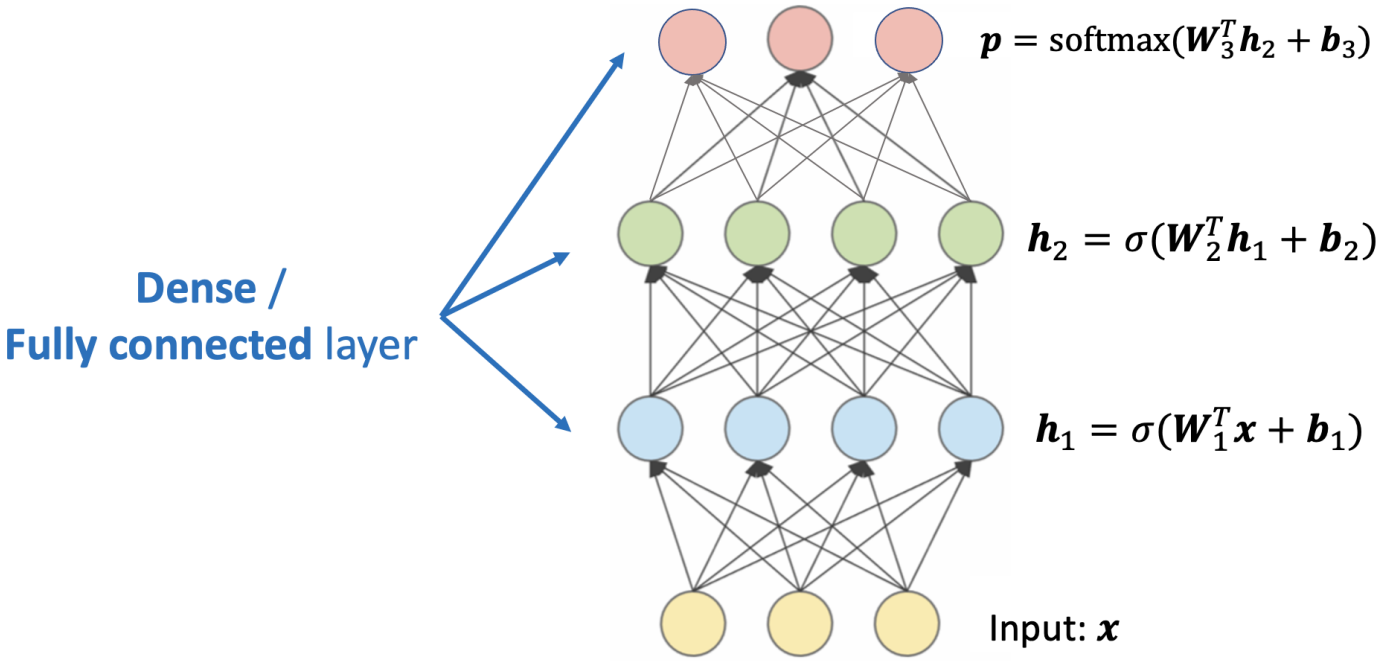
Define network architecture (1)

- Define a network as a Sequential object
- Add layers to it one-by-one

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(50, activation='relu'))
model.add(Dense(50, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

Define network architecture (2)



One-hot encoding

id	color
1	red
2	blue
3	green
4	blue

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Labels in Keras are usually encoded as one-hot vectors

Demo: train an MLP using Keras

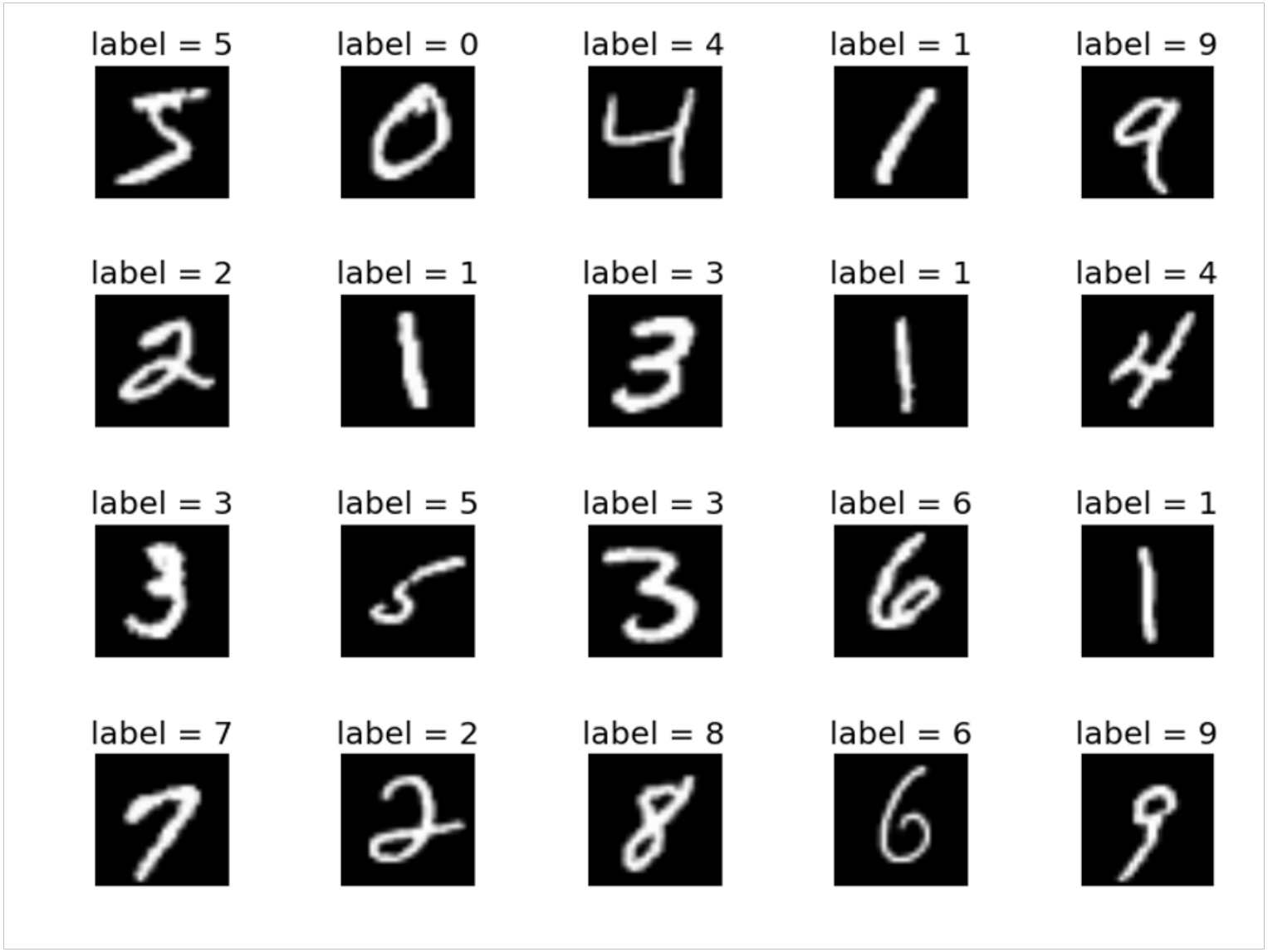
Some intros to computer vision

Computer vision

A field that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs

- Image classification/object recognition
- Object detection
- Image segmentation
- Image generation
- Image style transfer

Image classification



Object detection

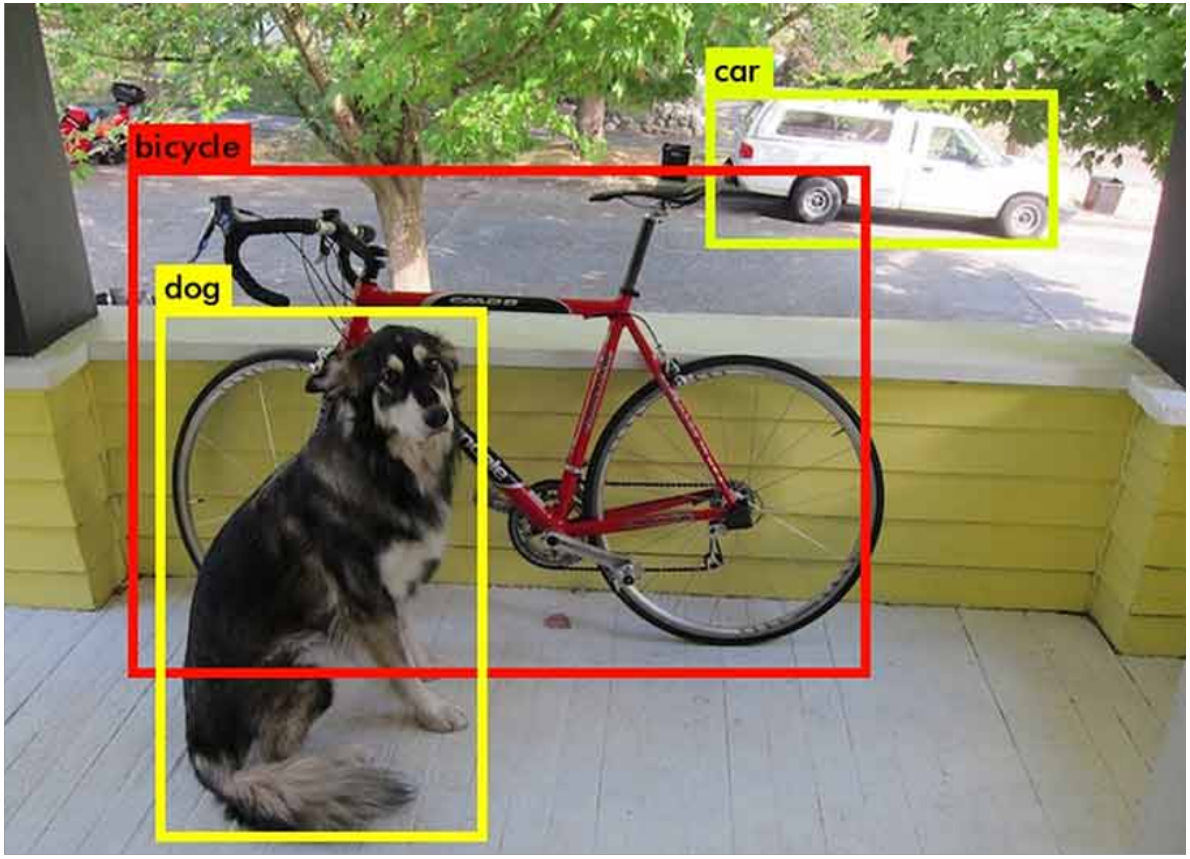


Image segmentation

Image generation

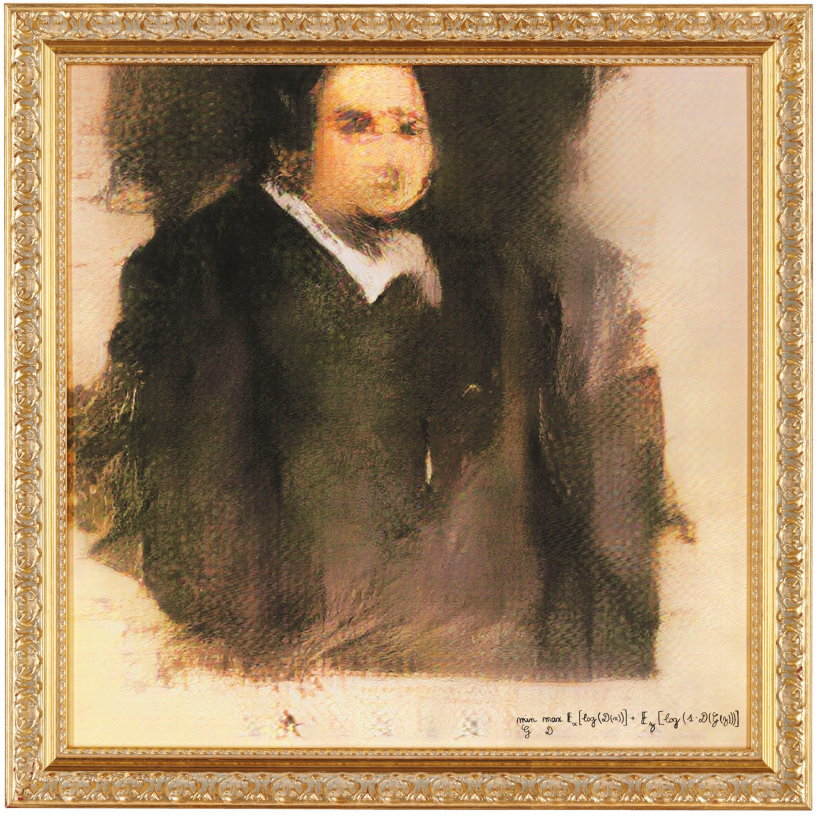
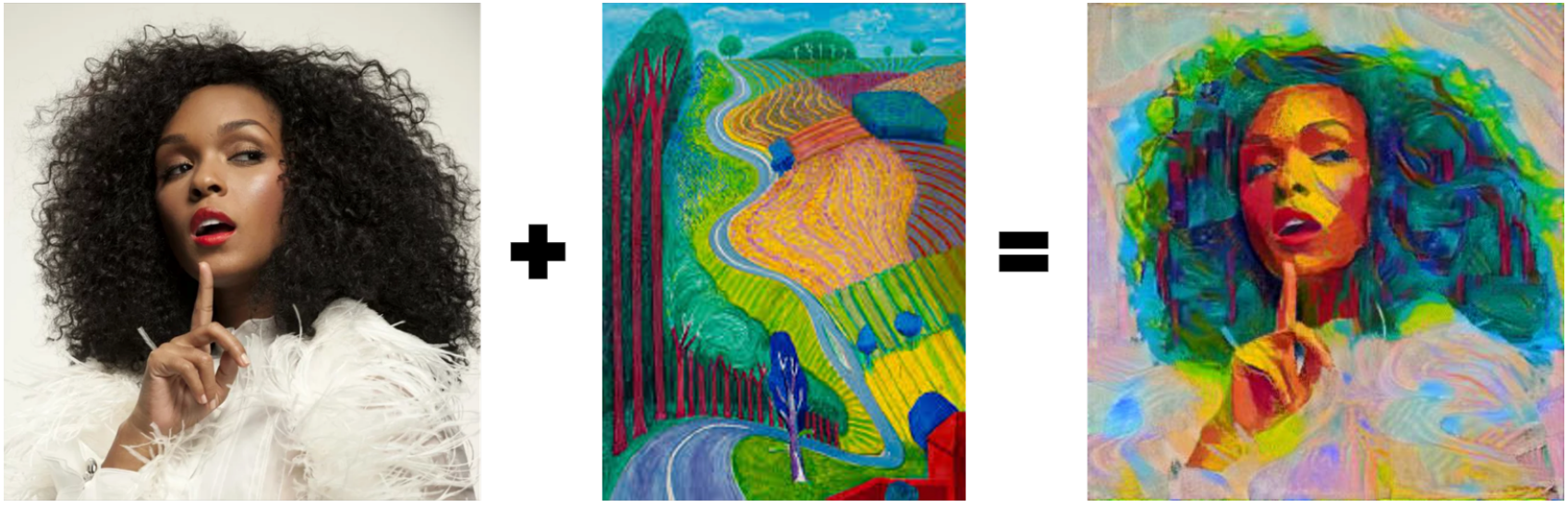
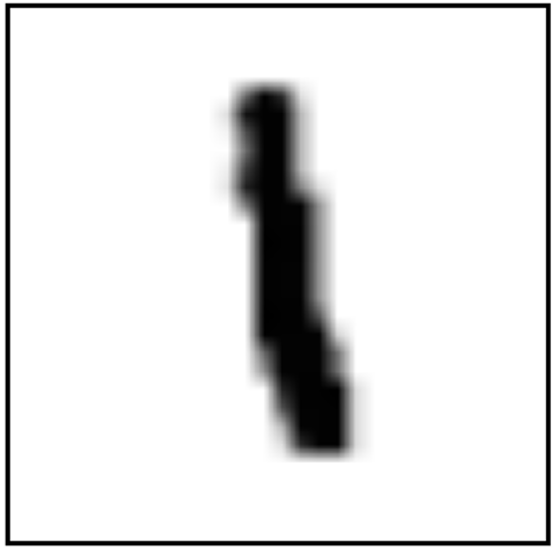


Image style transfer



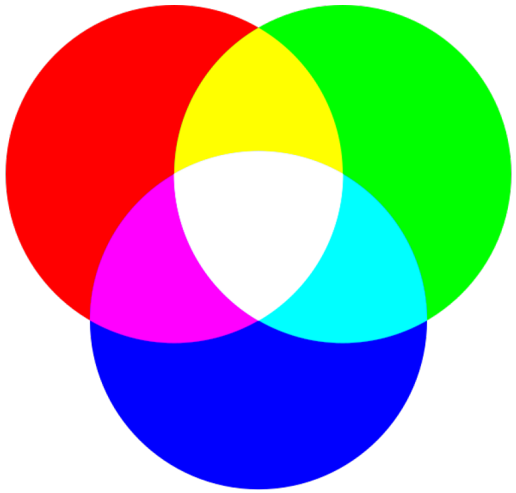
Grayscale image representation



$\approx$

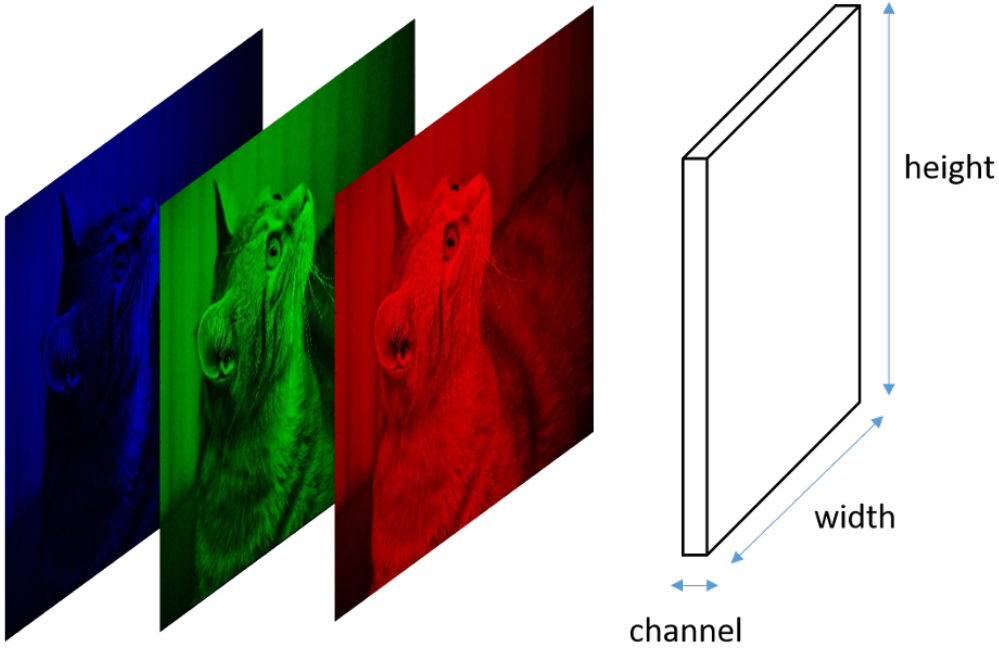
$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .6 & .8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .5 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .7 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .9 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .3 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Color image representation



- Use RGB color mode
- Represent a color by 3 values: R (Red) G (Green) B (Blue)
- There are other color modes

Image representation



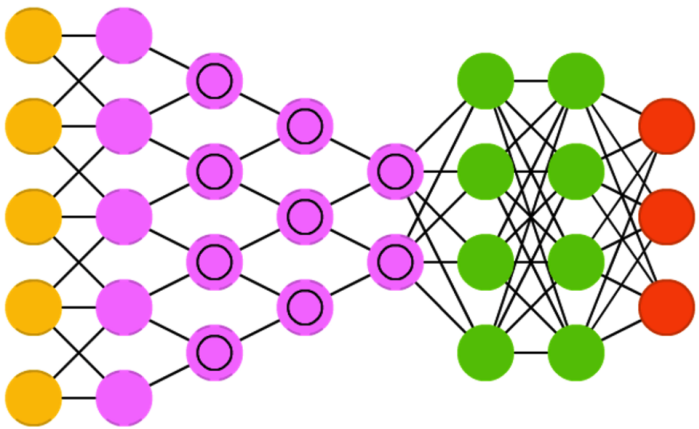
- An image is an H x W x C matrix: H (height), W (width), C (depth or number of channels)
- Grayscale image: $C = 1$
- RGB image: $C = 3$

Convolutional neural networks

Issues with MLP

- Flat vectors lose spatial information
- Sensitive to location of the object
- Cannot capture small regions within an image
- Redundant parameters

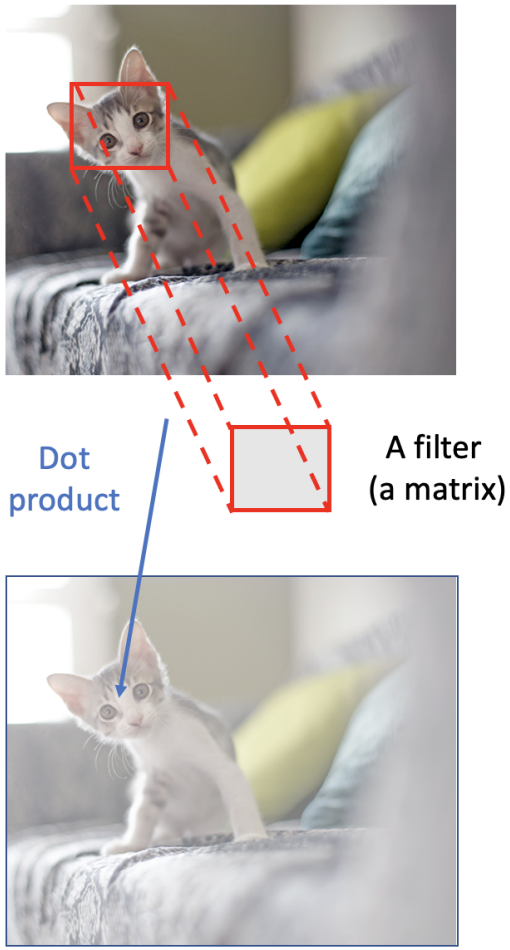
Convolutional neural networks



- two main parts: feature extraction and learning
- formed using 3 types of layers:
  - Convolution
  - Pooling
  - Dense layers

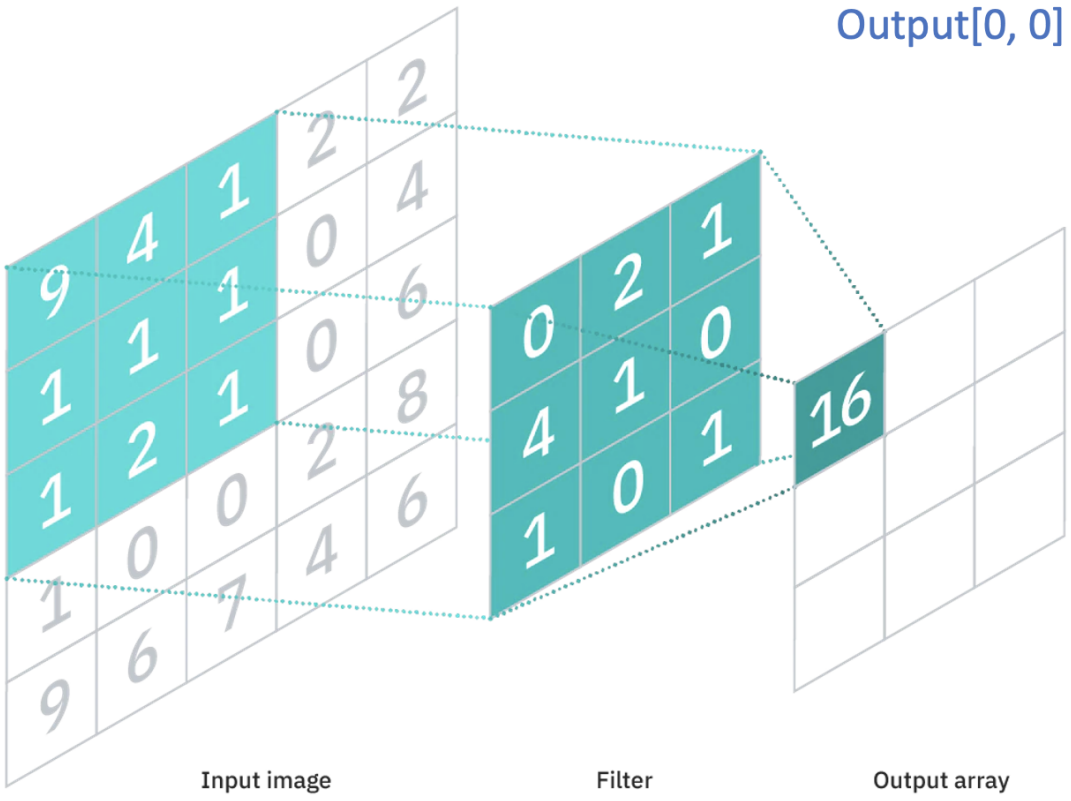
Convolutional layer

Convolutional layer



- Do not flatten the input image
- Apply a filter (kernel) to each local region of the image
- Slide the filter through all spatial locations to get the output

Applying a kernel



Output[0, 0] = (9*0) + (4*2) + (1*1) +
(1*4) + (1*1) + (1*0) +
(1*1) + (2*0) + (1*1)
= 16

(Adapted from IBM)

Examples of kernels

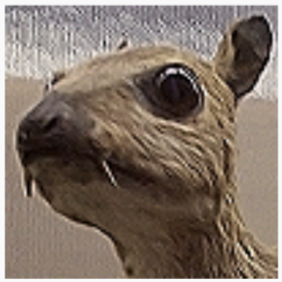
Edge detection

Kernel

$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

Sharpen

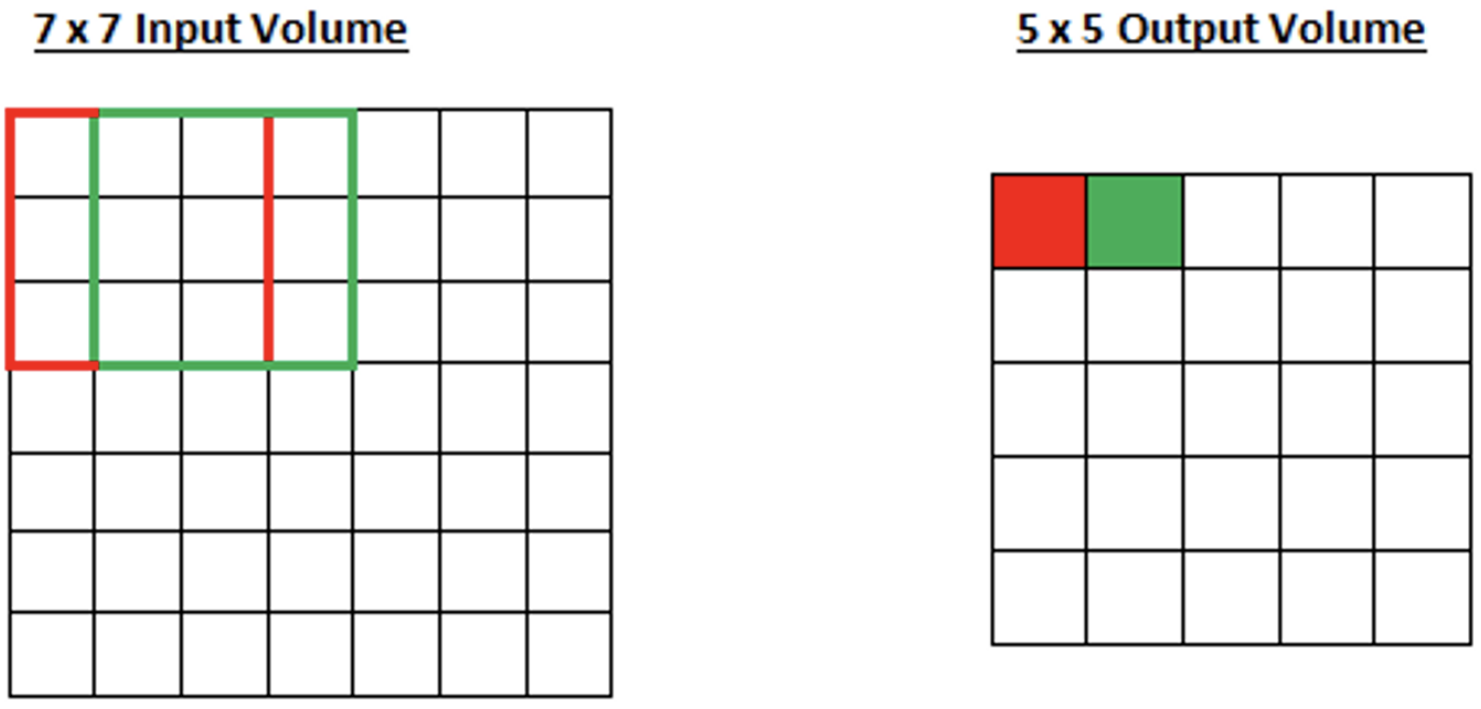
$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$



Feature extraction

- Historically, convolutional filters have been used to extract image features
- CNNs automate that process by considering the entries of the filters as model parameters

Applying a kernel



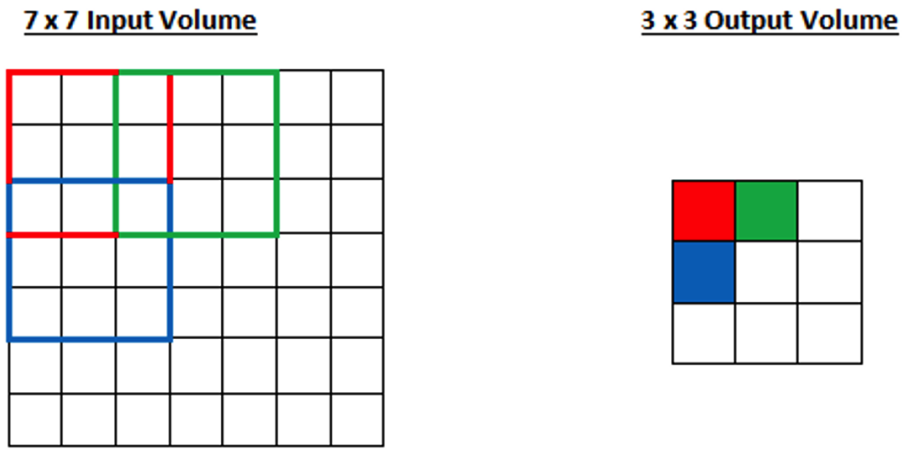
Applying a kernel

https://poloclub.github.io/cnn-explainer/assets/figures/convlayer_detailedview_demo.gif

Stride

- Number of pixels to shift the filter
- Can be different for each dimension

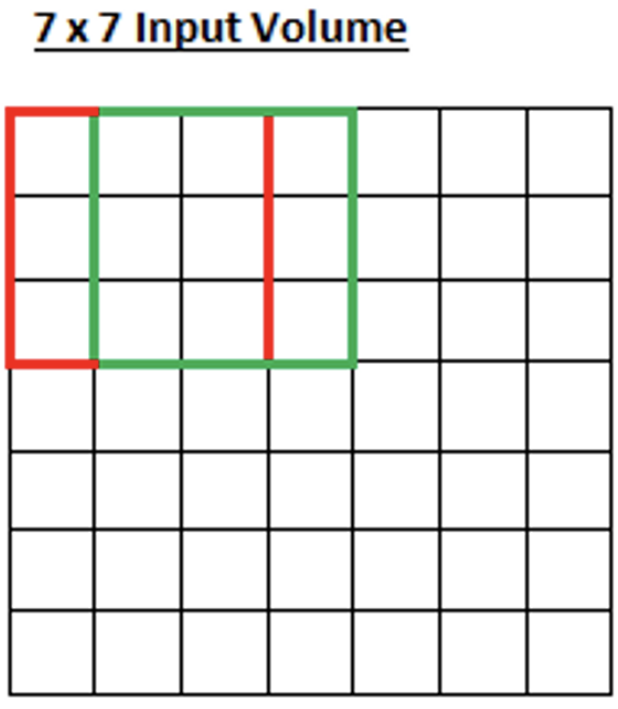
strides = (2, 2)

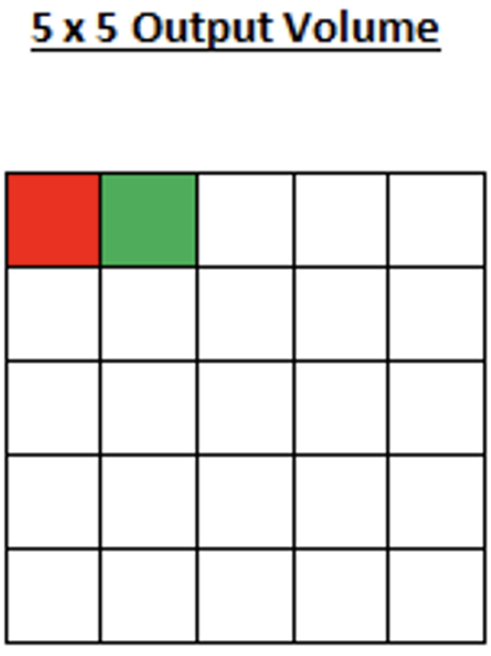


(Source: Adit Deshpande)

Padding

2 possible settings: Valid or Same

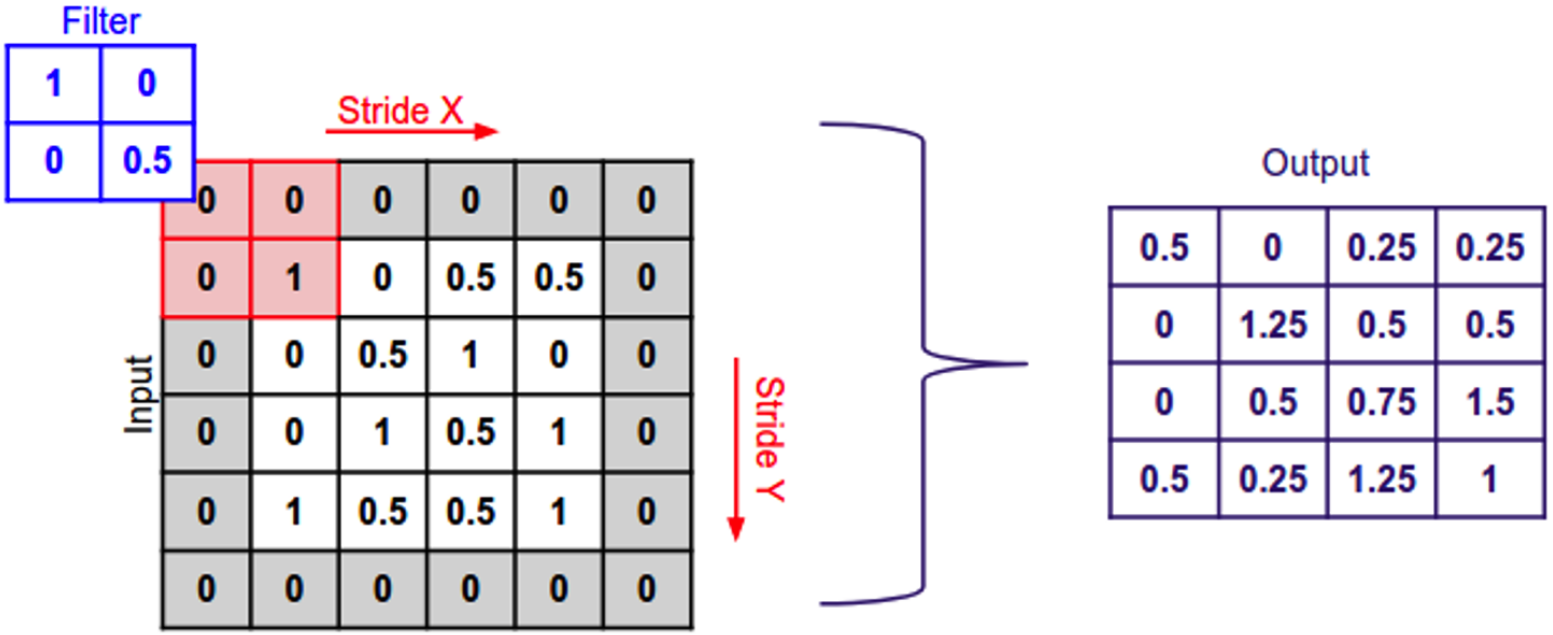




(Valid)

Padding

Same padding: add 0 pixels to boundary of input image to get similar output shape



(Same)

Applying a kernel

https://vucdinh.github.io/Presentation/figures/cnn/filter-run.gif