A stroll through Computer Vision

Benjamin Kiessling

September 29, 2023

Introduction

The purpose of this lecture is to give you the ability to translate humanities problems into computer vision terminology.

By the end you should be able to independently select one or more classes of methods and algorithms for your image processing needs even if you don't understand the methods in their entirety.

Introduction - Computer Vision?

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

Artificial Intelligence Group July 7, 1966
Vision Memo. No. 100.

THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

Low Level: Resizing

Low Level: Image Adjustments

Low Level: Grayscale

Low Level: Exposure

Low Level: Saturation

Low Level: Edges

Low Level: Binarization

Low Level: Segmentation

Low Level Vision

Photo manipulation such as:

- size
- color
- exposure

Feature extraction:

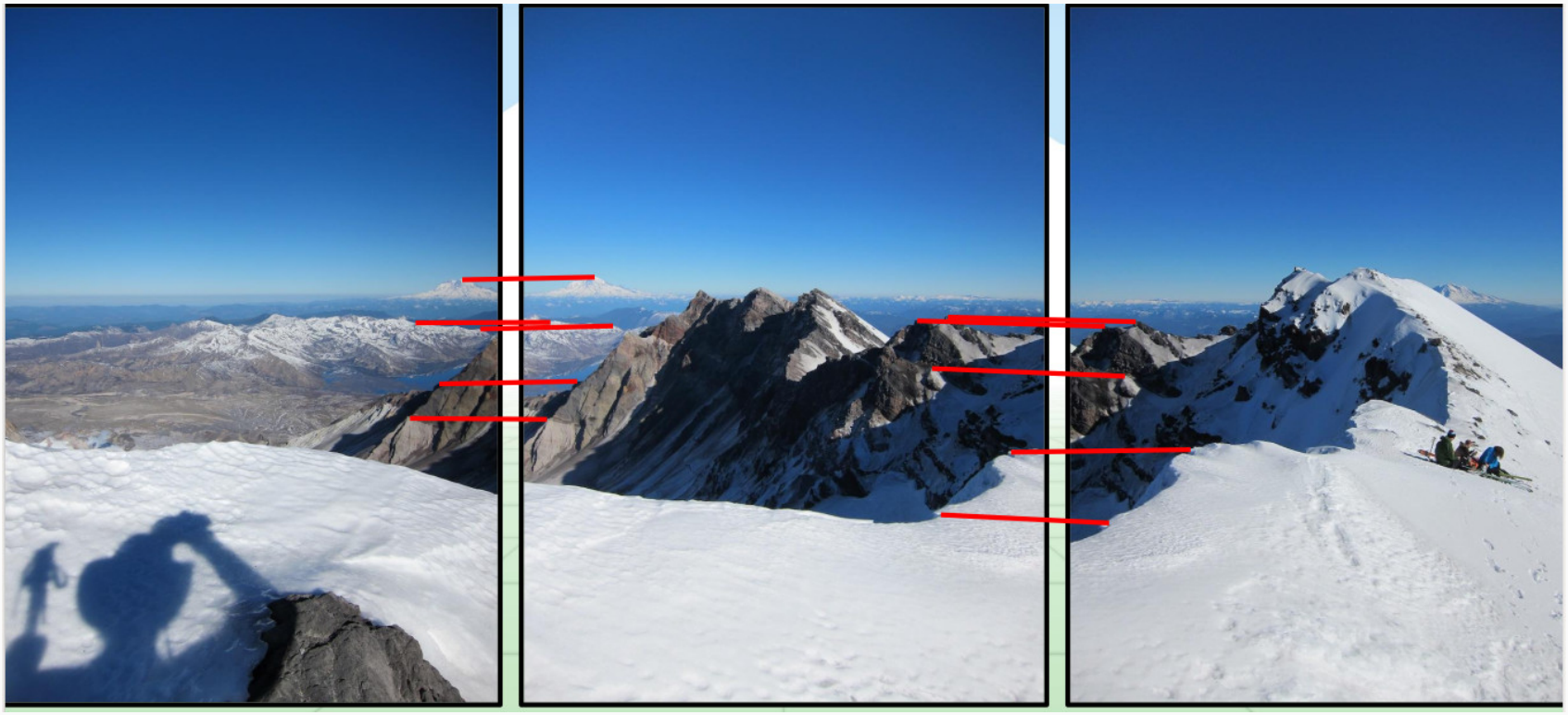
- edges
- oriented gradients
- segments

Low Level Vision - Applications

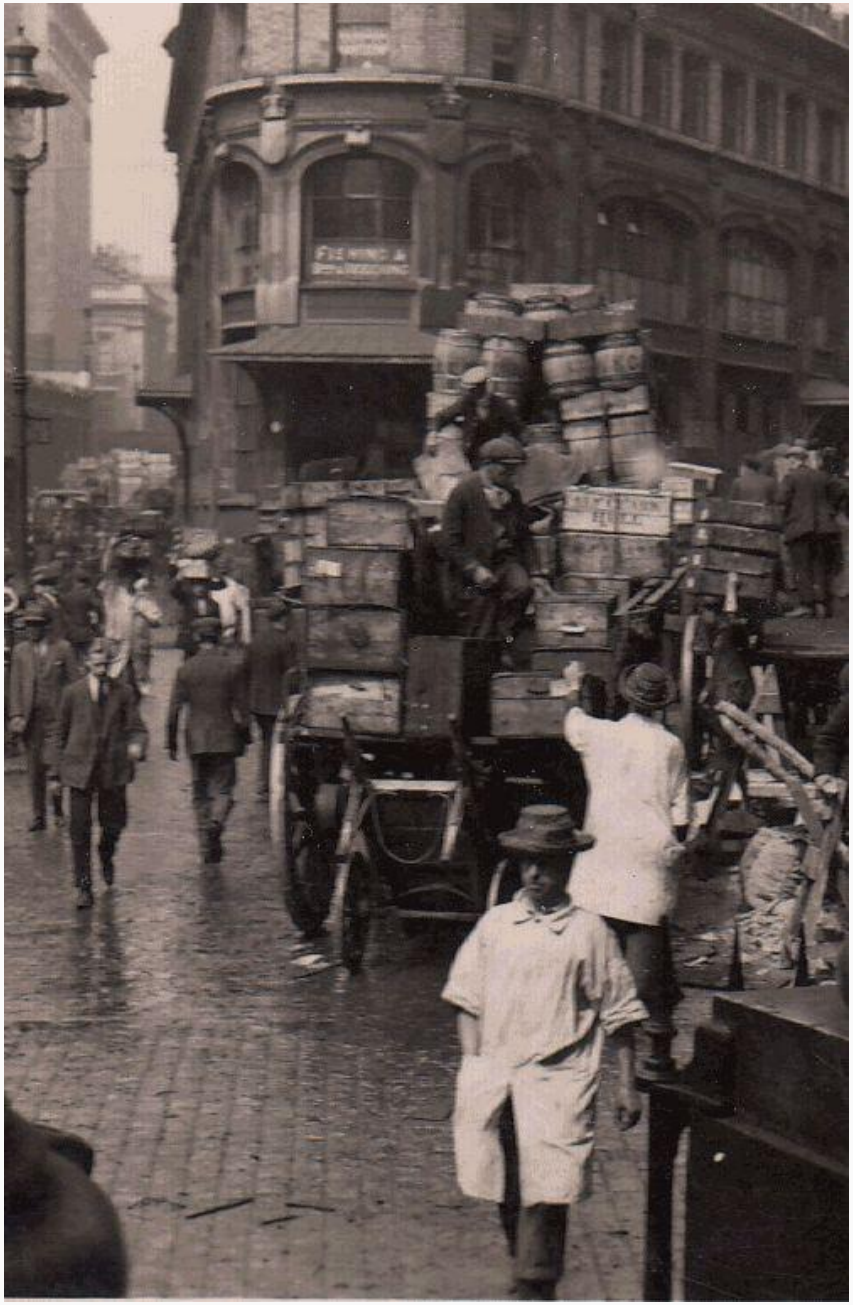
Largely making material accessible to human interpretation.

Sometimes preprocessing in higher complexity applications.

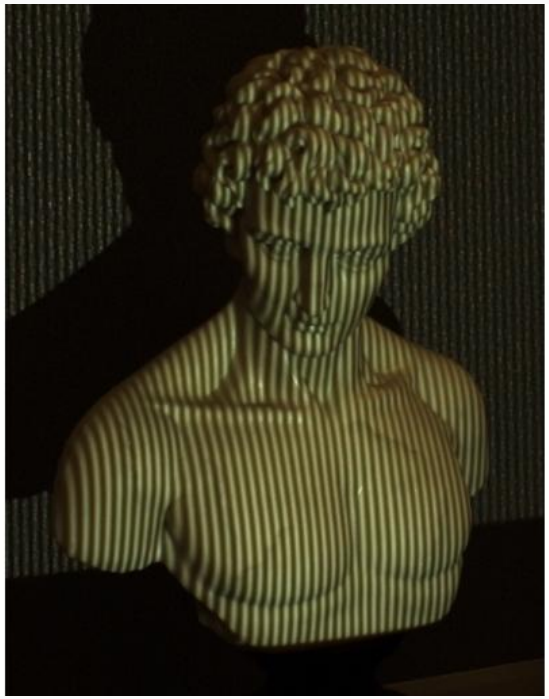
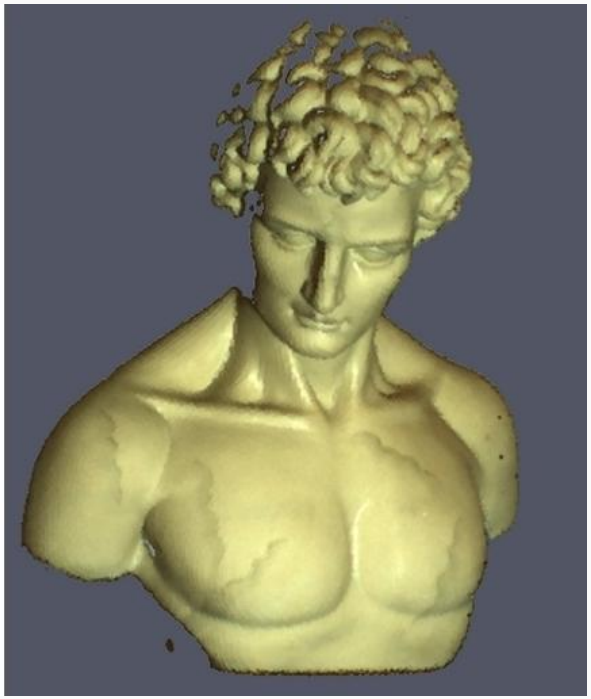
Mid Level: Panorama Stitching



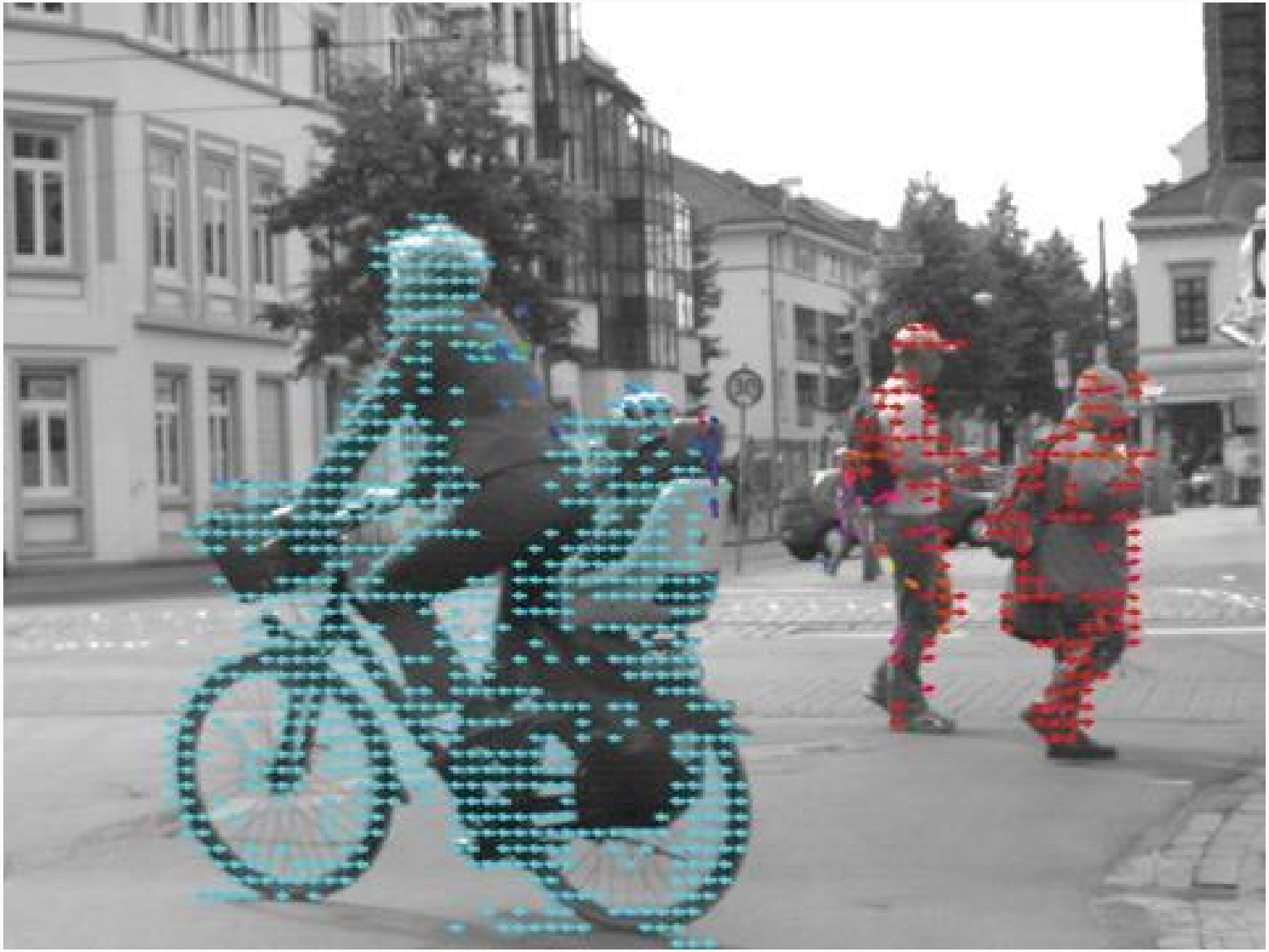
Mid Level: Multi-View Stereo



Mid Level: Structured Light Scan

Mid Level: Optical Flow



Mid Level: Time Lapse

Mid Level Vision

Image to image

- panoramas
- seam carving

Image to world

- structure from motion
- structured light
- LIDAR

Image in time:

- optical flow
- time lapse

High Level Vision

High Level: Classification

What is in the image?

High Level: Tagging

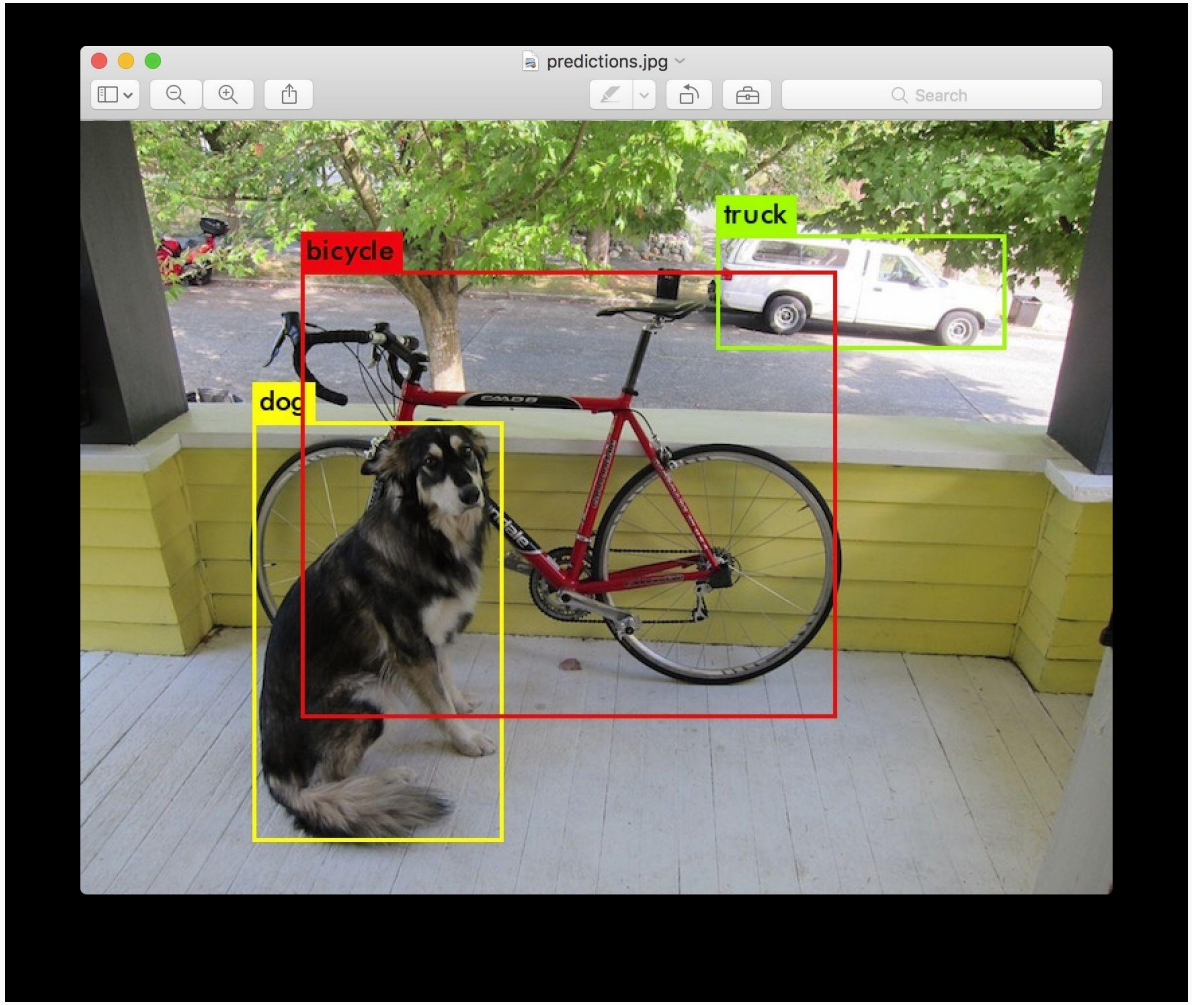
What are all the things in the image?

Classification with more than one answer.

High Level: Object Detection

What are all the things in the image?

Where are they?



High Level: Semantic Segmentation

What type of thing is each pixel?

High Level: Instance Segmentation

Object Detection + Segmentation

High Level: Panoptic Segmentation

Object Detection + Semantic Segmentation

High Level...

Single image 3D

Game playing

Super-resolution

Retrieval

Whatever problem some researcher made up last week because they needed an application for their method.

High Level Vision

Semantic!

- image classification
- object detection
- segmentation

Applications:

- autonomous cars
- facial recognition
- ...

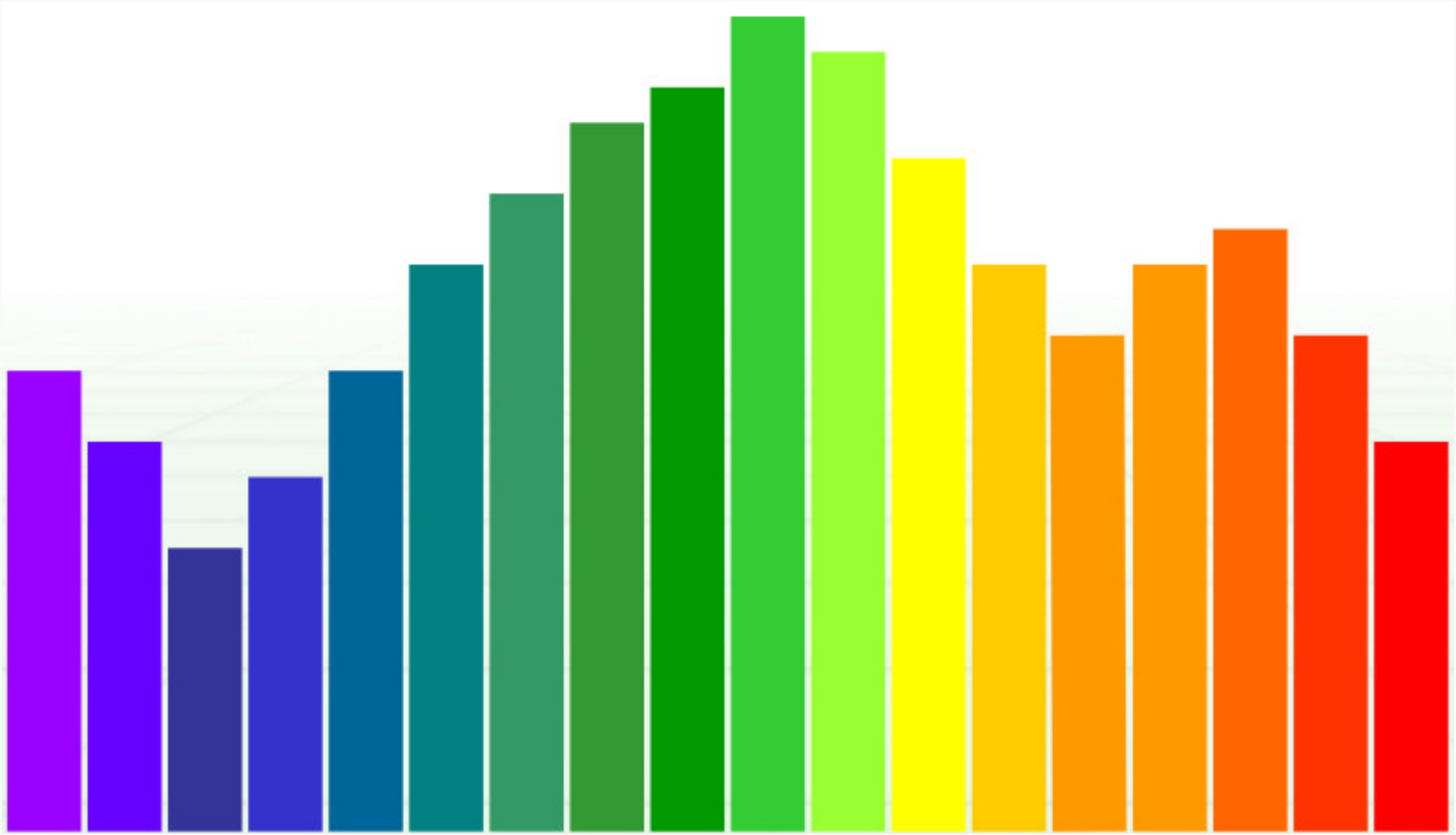
but also:

- text recognition
- reassembling fragmentary material
- dating

Low Level: Light and Color

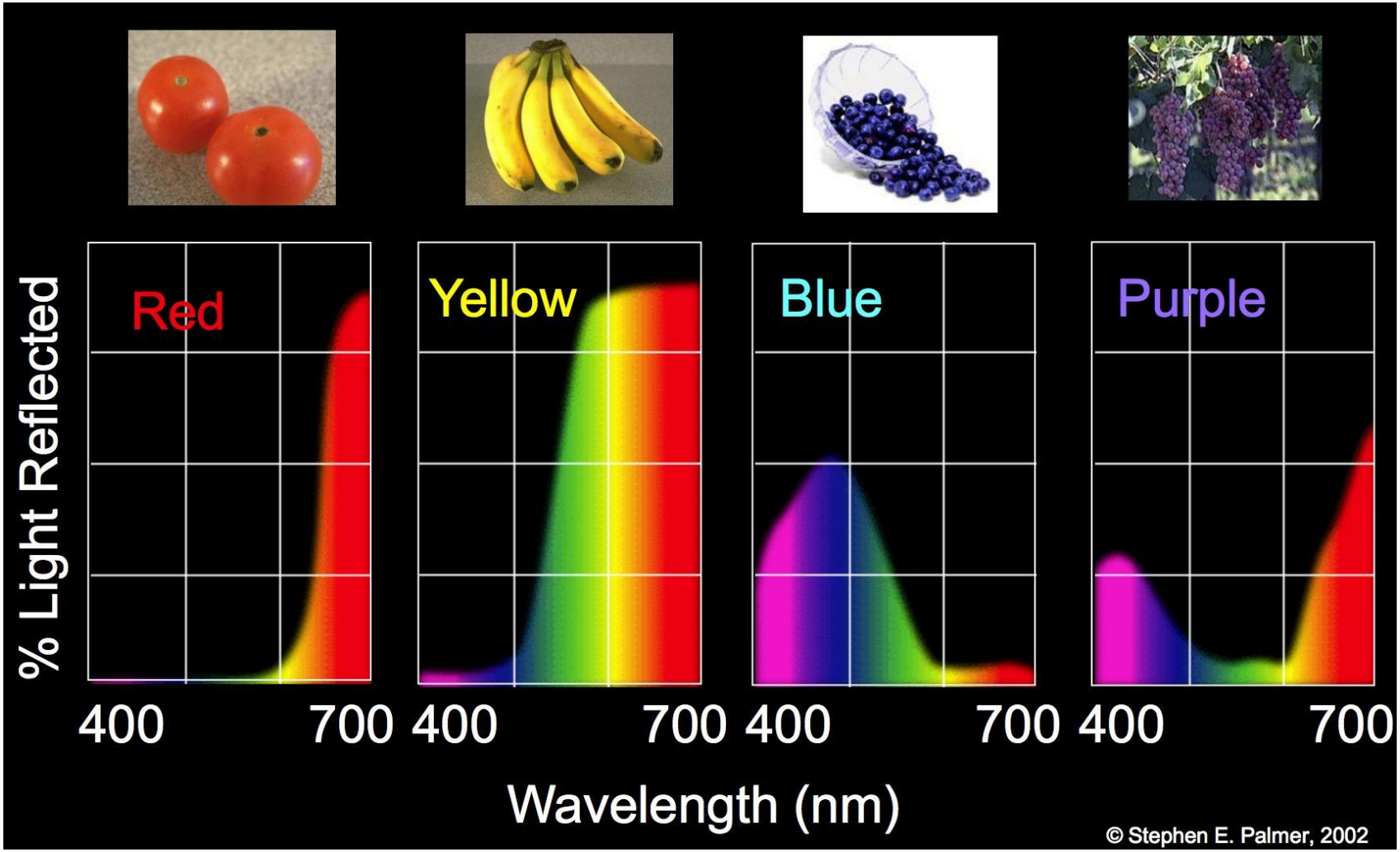
Light is a combination of waves, like a chord in music.

It can be described as a sum of its parts.



Low Level: Light and Color

Objects reflect only some light.



Low Level: Light and Color

Our eyes contain different photoreceptors (120 million rods and 6 million cones).

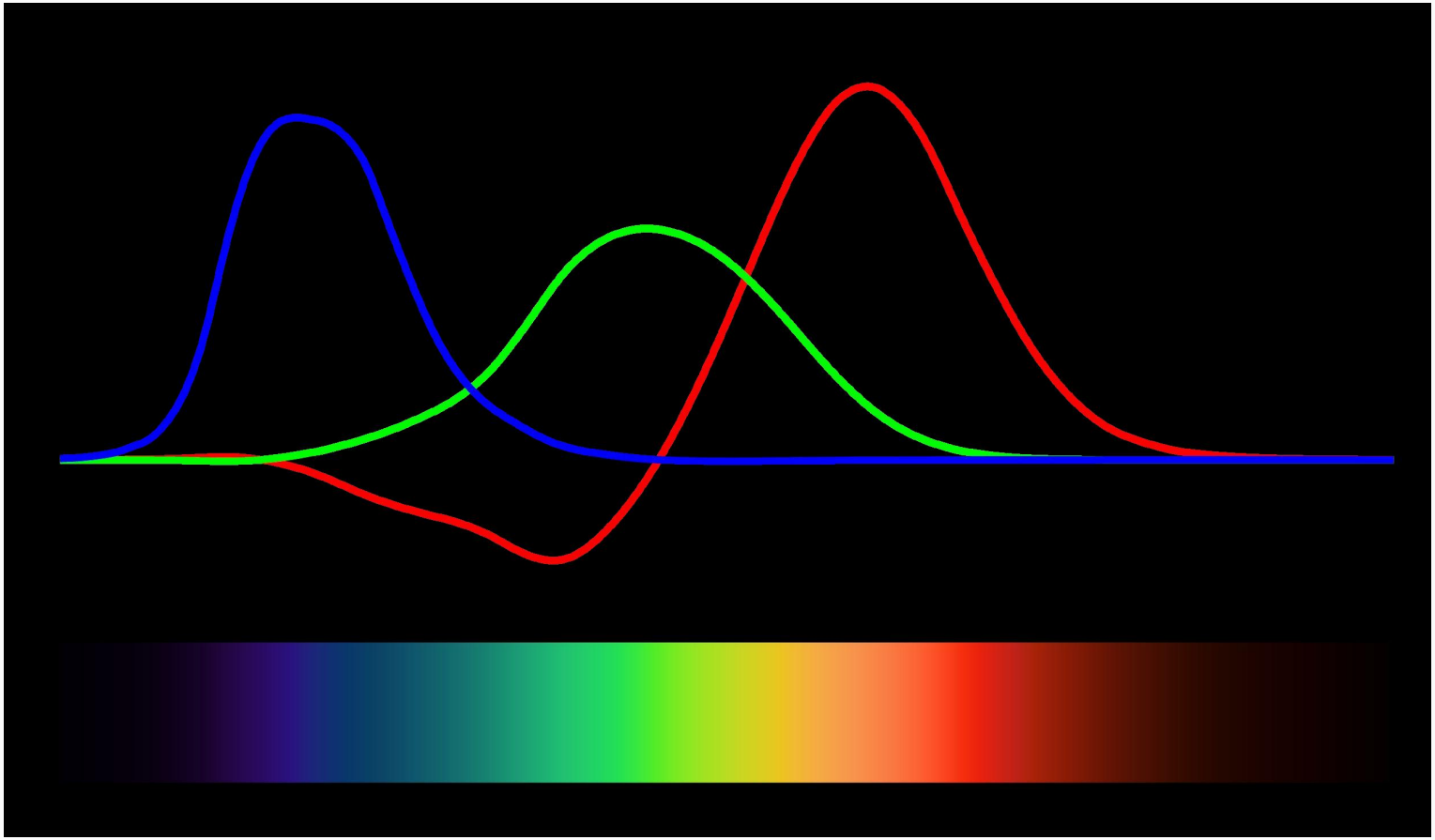
They are more responsive to some wavelengths of light than others.

Rod are responsible for grayscale vision.

Cones are responsible for color vision with 3 separate types for red, green, and blue vision. They are not equally distributed!

Low Level: Light and Color

Color is our perception of waves!

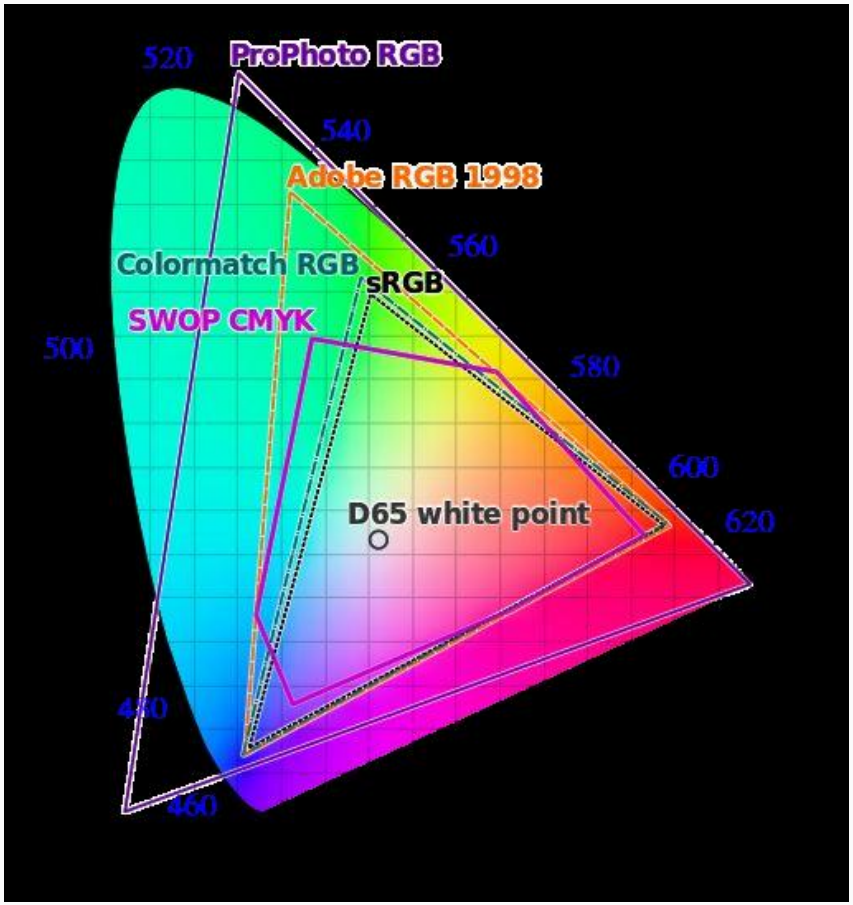


Low Level: Light and Color

Computers represent images as grid of pixels.

Each pixel has a color, 3 components: RGB.

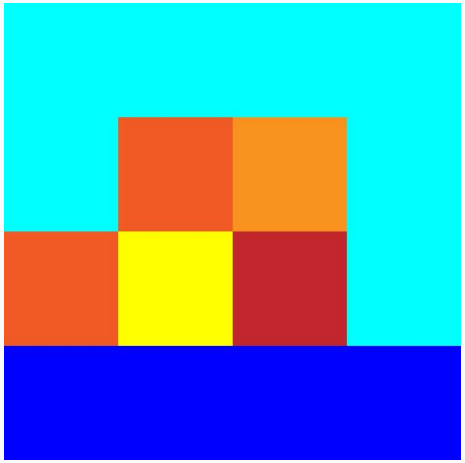
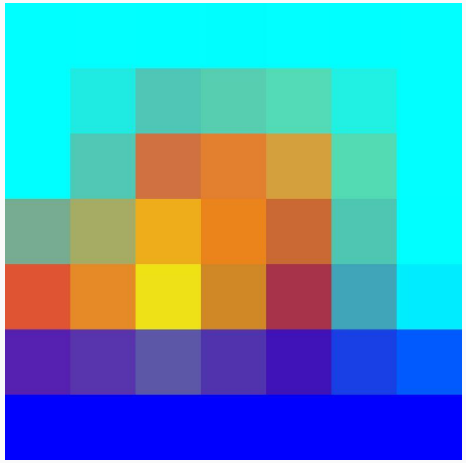
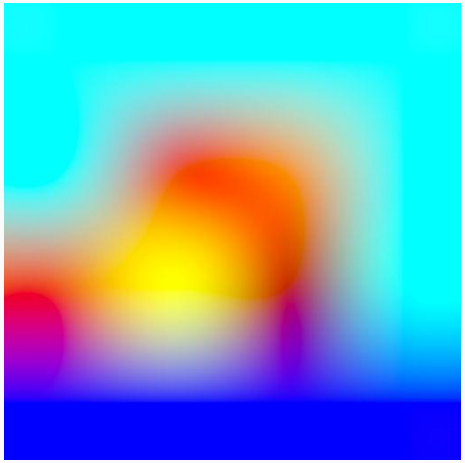
Not every color can be represented in RGB! RGB is made to trick humans, not be accurate.



Low Level: Image interpolation and resizing

To increase the resolution of an image we interpolate the points between pixels of the source image.

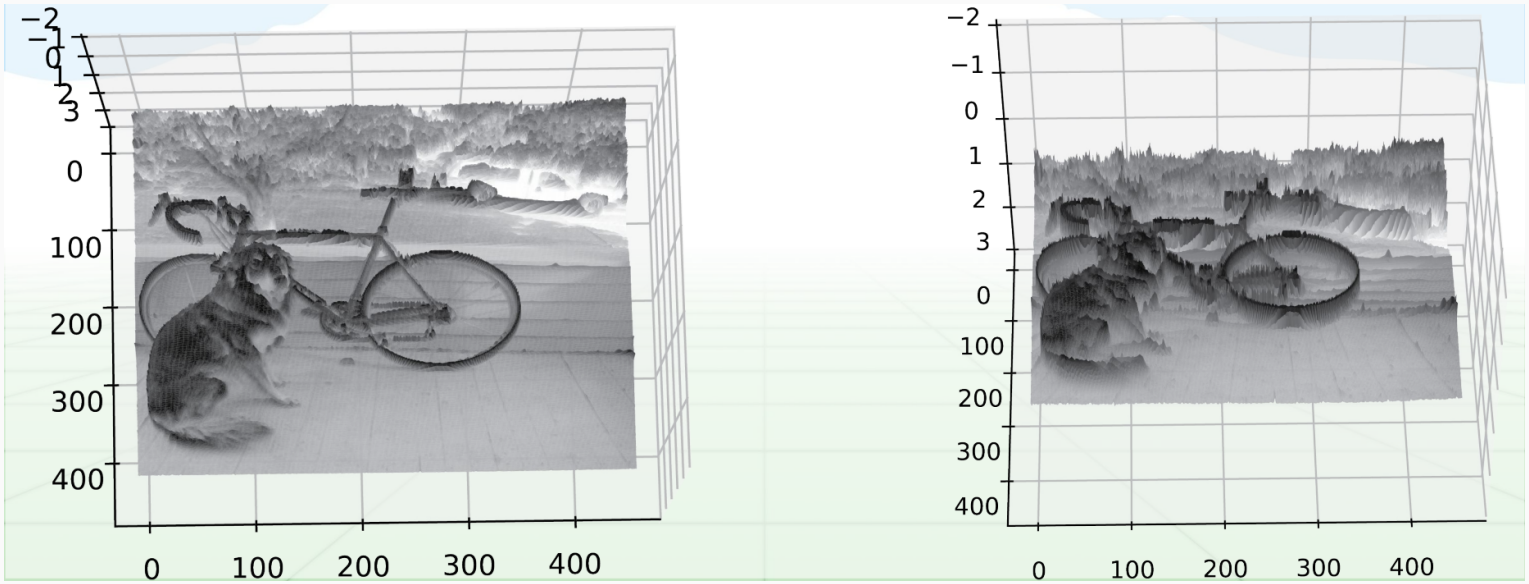
The new pixels are a function of adjacent original pixels. There are different interpolation algorithms (linear, bilinear, cubic, bicubic, Lanczos) with the usual tradeoff being speed vs quality.

Low Level: Edges and Features

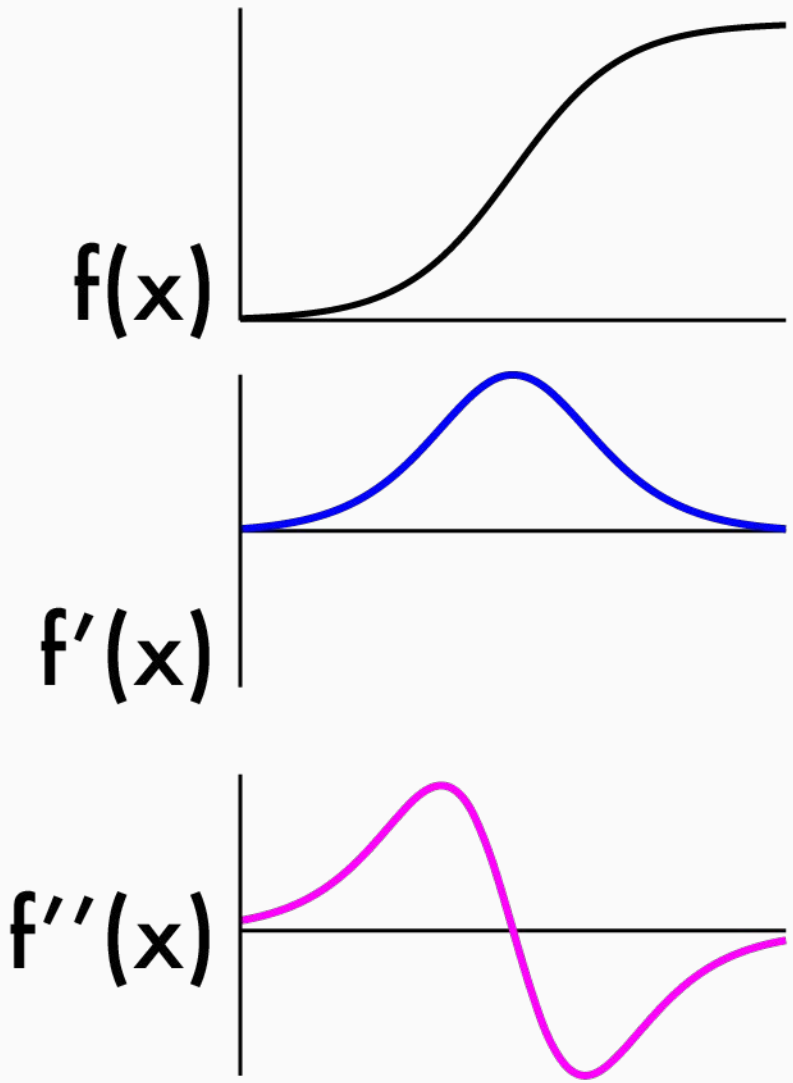
Image is a function

Edges are rapid changes in this function



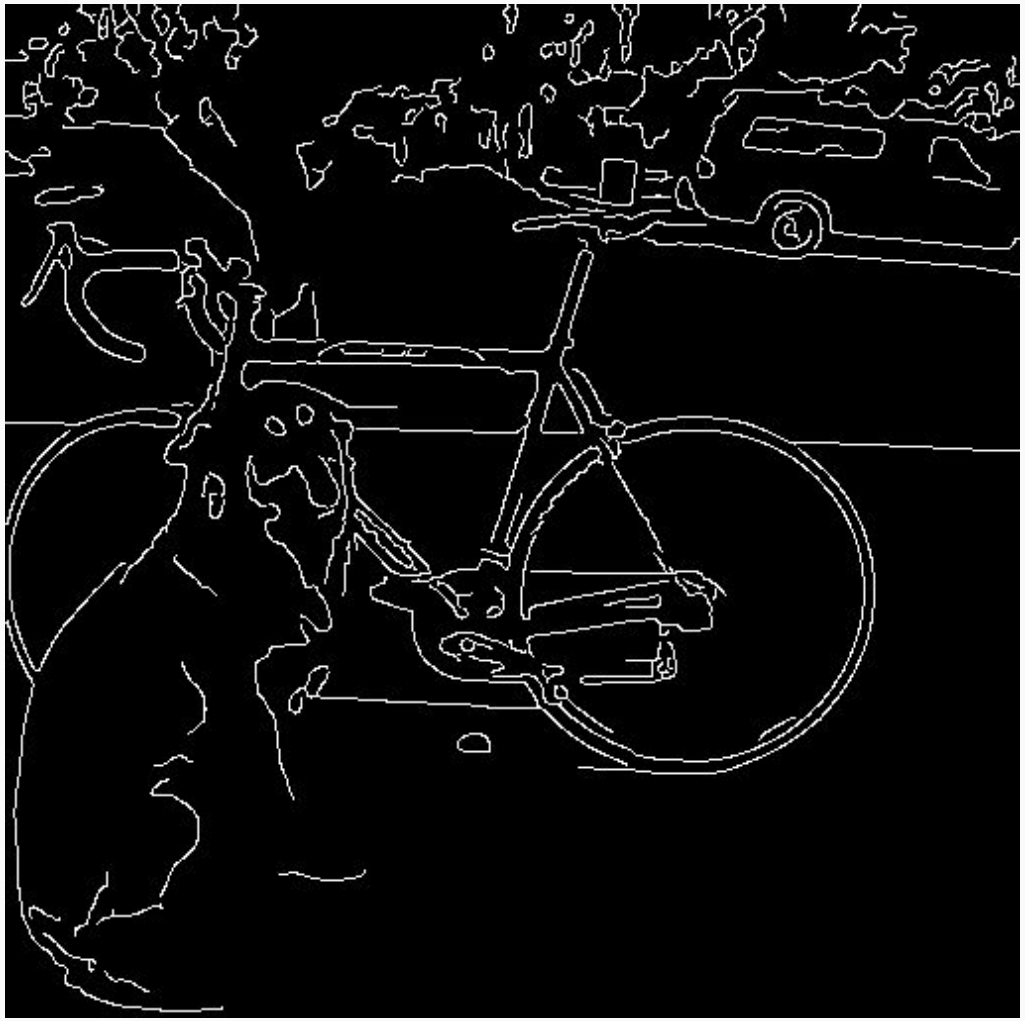
Low Level: Edges and Features

Image is a function.
Edges are rapid changes in this function.



Mid Level: Line Drawing

Usually we don't really want edges but line drawings!



Mid Level: Canny Edge Detection

Not a single function but a pipeline! Because we suck at doing anything in one step we chain multiple operations.

1. Smooth image
2. Calculate gradient
3. Non-maximum suppression
4. Thresholding into strong, weak, and no edge classes
5. Connect components.

No need to implement, is part of any self-respecting low level computer vision library.

Mid Level: Features

Features are highly descriptive local regions and a way to describe those regions.

Useful for matching, recognition, detection, ...

Mid Level: Features

Good features are distinctive of patches in an image that are useful or have some meaning.

For objects, a patch that is common to that object but not in general.

For panorama stitching we want patches that we can find easily in another image of the same place.

Mid Level: Features

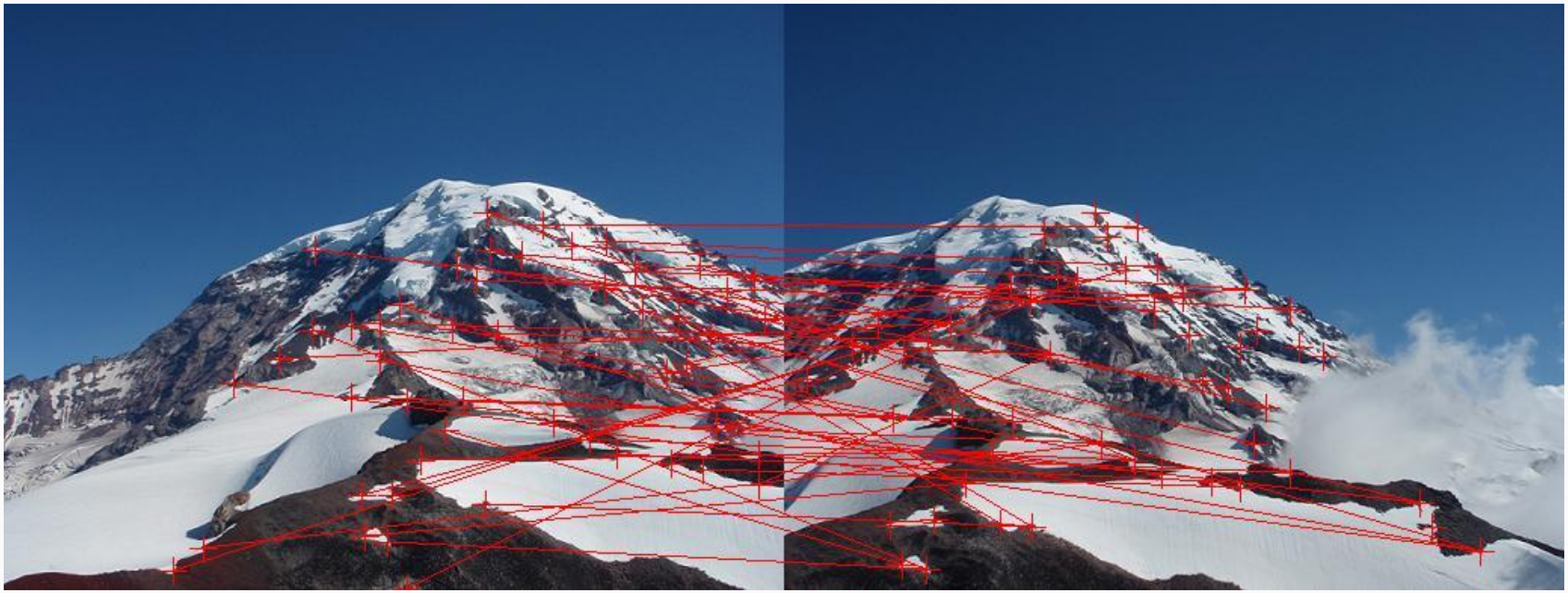
Sky: bad (very little variation).

Edge: ok (Could match other patches along same edge)

Corners: good (only one alignment matches).

Mid Level: Matching

Use patches around distinctive areas and compare!



Pause

High Level: Machine Learning

ML are algorithms to approximate functions. They usually minimize some form of loss function.

Can be classified into 3 clades of methods:

Supervised learning Given inputs to a function, predict the output. Have lots of labelled examples.

Semi-supervised learning Same but number of labelled examples is smaller than the number of examples.

Unsupervised learning Modelling unlabelled data. Find similarities and differences between subgroups of data.

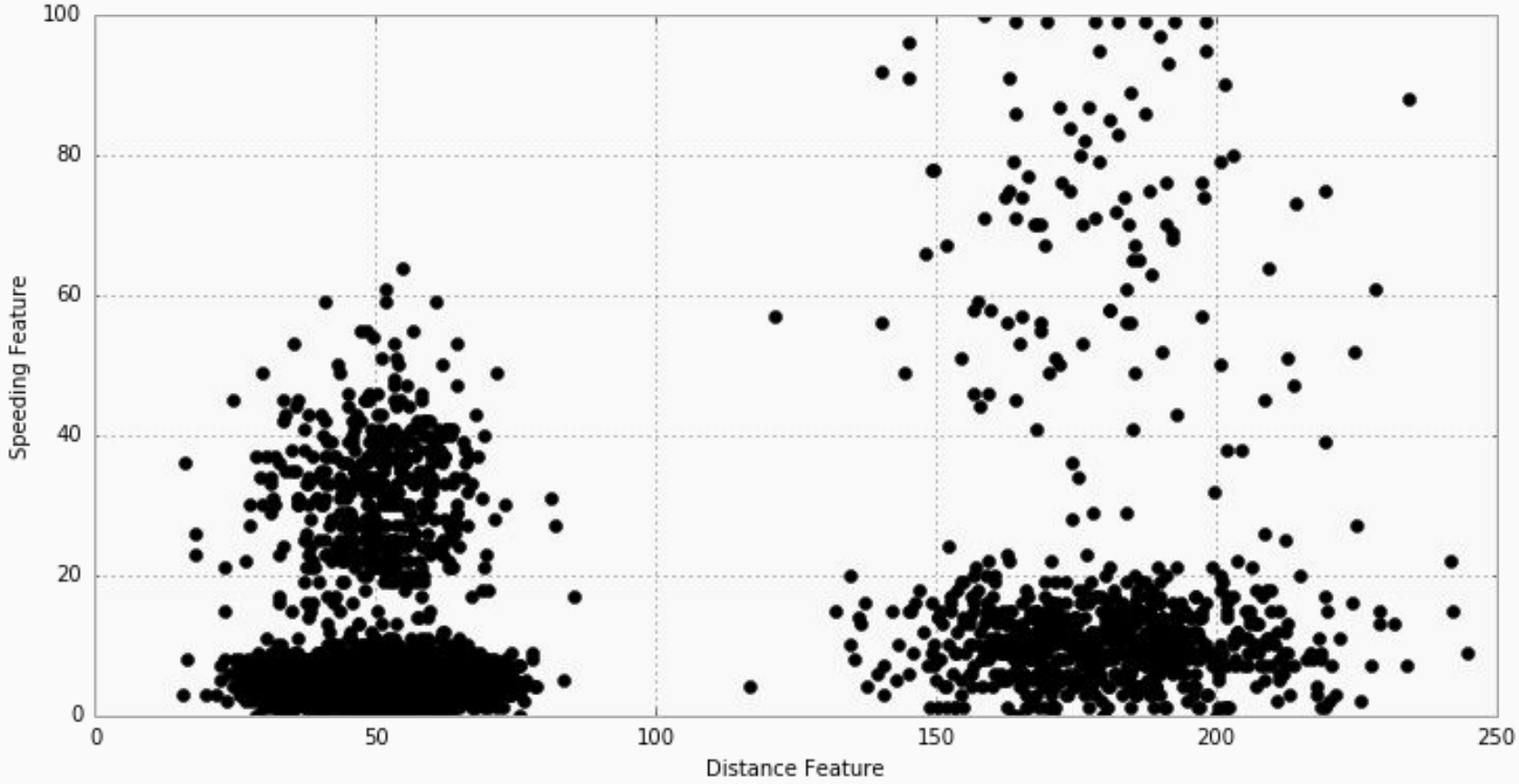
Unsupervised learning

No labels, just looking for patterns in data.

The most widespread unsupervised learning methods are clustering algorithms.

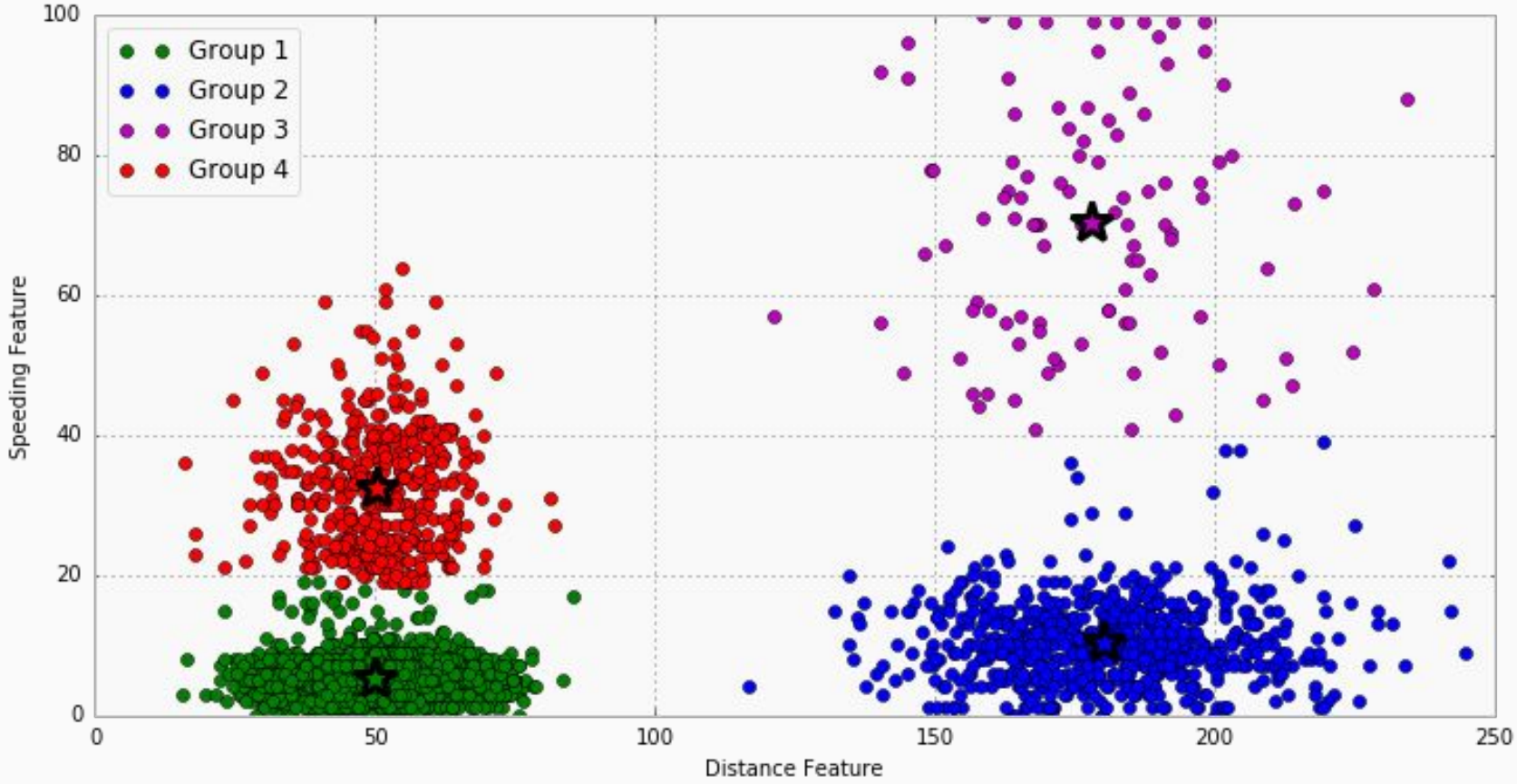
Unsupervised learning - Clustering

Finding groups in data.



Unsupervised learning - Clustering

Finding groups in data.



K-means clustering

Assume points are close to other points in group, far from points out of group.

Algorithm:

1. Randomly initialize cluster centers.
2. Calculate distance of all points to centers
3. Assign points to closest cluster center.
4. Move cluster center to average position of all assigned points.
5. Repeat!

Clustering on images

Group together pixels by color, automatic segmentation: k-means, k=2

Clustering on images

Group together pixels by color, automatic segmentation: k-means, k=4

Clustering on image data

We can cluster any data where a distance function between two points exists.

An unsupervised object detection system might compute features on patches and cluster them afterwards to find similar objects in images.

Supervised learning

We have data with labels, want to take new data and predict the correct label.

1. Map the desired labels to a problem, such as regression, classification, ...
2. Pick a model. Do we know anything about the data (is it linear, polynomial, high dimensional, ...)?
3. Pick a loss function. Depends on the label type.
4. Train and evaluate.

We skip over the training part here (look up gradient descent and backpropagation if you're interested).

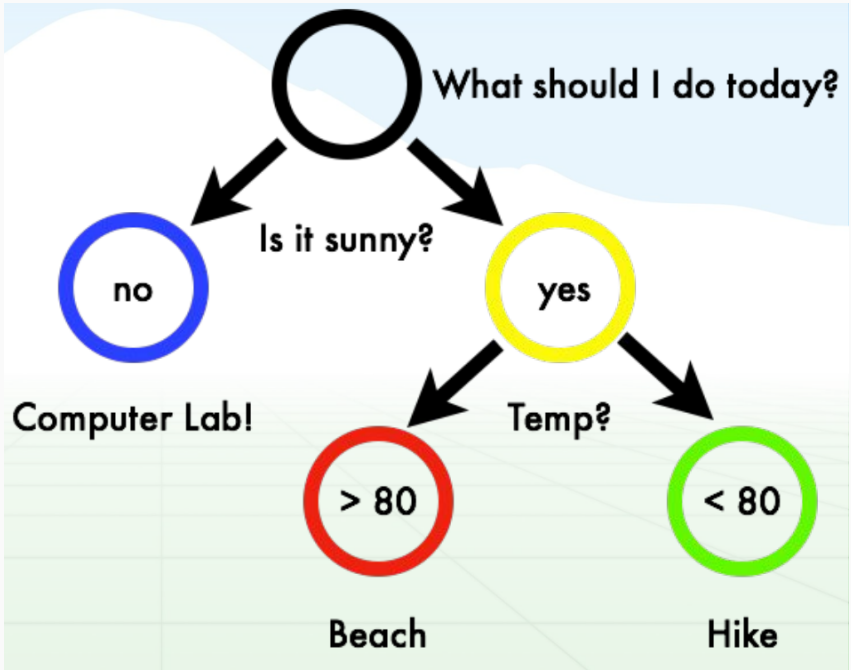
Supervised learning - Trees

Decision trees are very simple models.
Benefits:

- interpretable
- easy to use

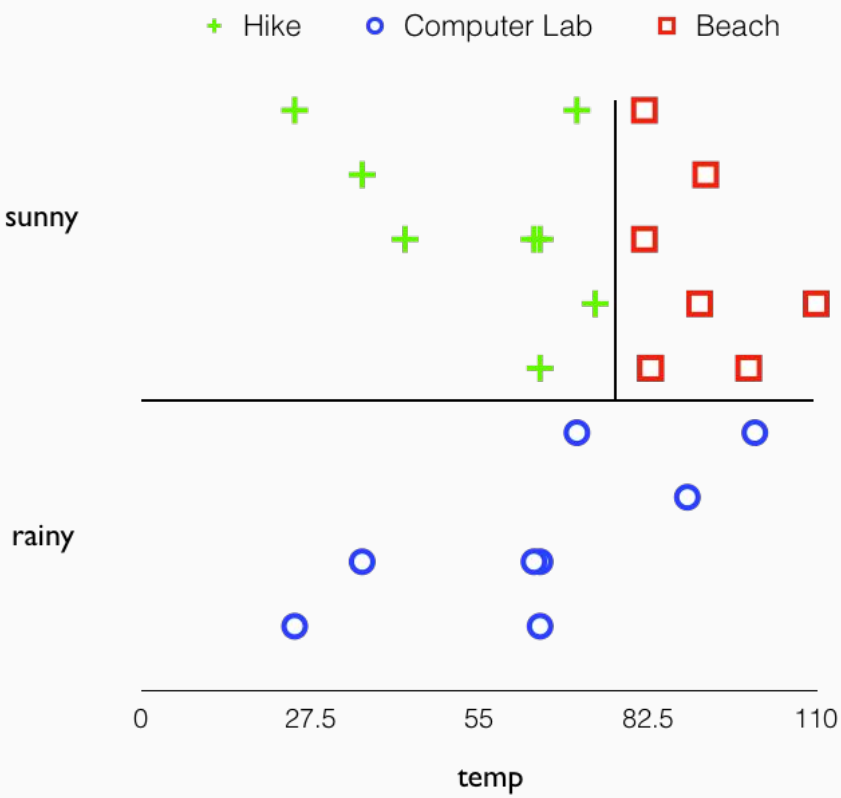
Drawbacks:

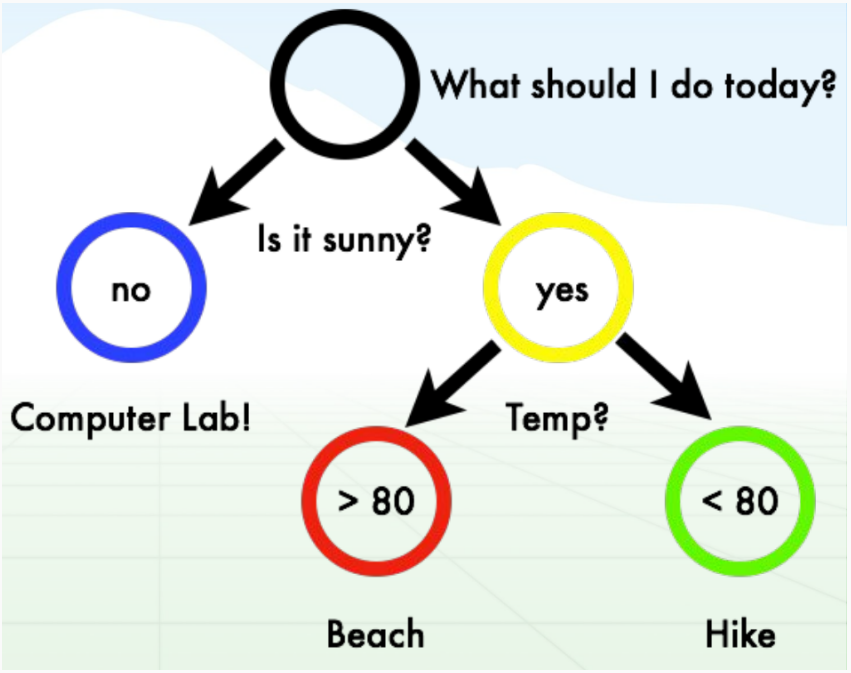
- are usually quite weak
- can require expertise in construction



Supervised learning - Trees

Trees are partition of data.





Supervised learning - Boosting

Boosting is a way to make a weak classifier better.

We start by training a weak classifier and make the data it got wrong more important (increasing its weight).

And retrain the classifier. And do that over and over again.

The final classifier is a combination of our weak classifiers!

Supervised learning - Cascades

Given a bunch of classifiers with known properties we can construct a cascade of classifiers.

1. Very fast, throws out easy negatives.
2. Fast, throws out harder negatives
3. Slower, throws out hard negatives

Only run slow good classifiers on hard examples.

Supervised learning - Softmax

Regressions model the relationship between an input and output variable.

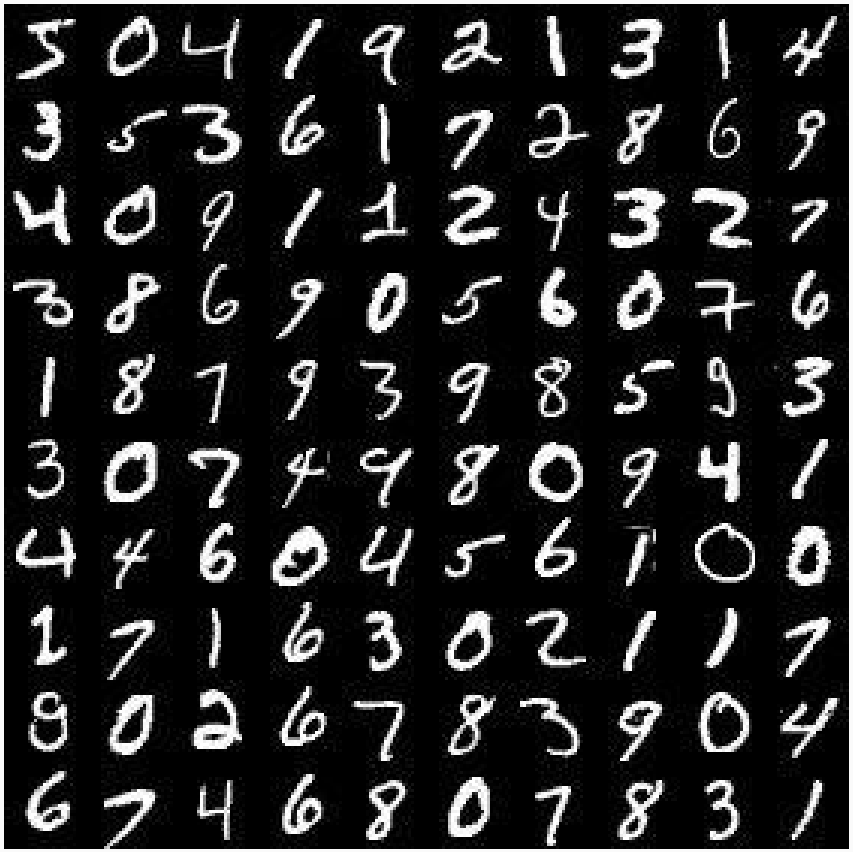
Logistic regression output a value between 0 and 1 for a given input.

These are usually interpreted as class probabilities.

Can be extended to multiple classes, it is then called multinomial logistic regression or softmax. Assigns a joint probability across classes.

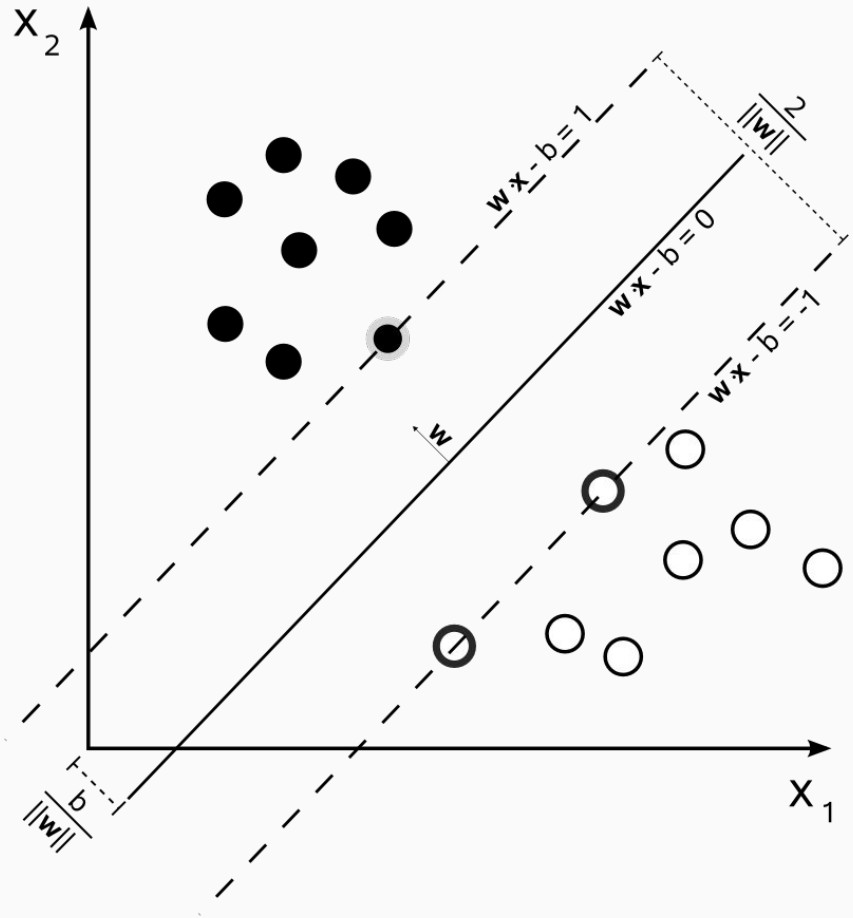
Supervised learning - Softmax examples

MNIST dataset of 50000
handwritten numerals.
10 classes with the raw pixel values
as inputs for the softmax regression
$>$ 95% accuracy



Supervised learning - Support Vector Machines

Workhorse of old-school vision algorithm. Finds the best linear classifier separating the data into two.
Better than logistic regression for high dimensional data (but still needs features to do most tasks).



Supervised learning - Neural networks in 2 slides

All these methods require features. Finding good features (feature engineering) is annoying.

Why not let the algorithm decide?

A neural network is a feature extractor + linear model (regression).

Supervised learning - Neural networks in 2 slides

The feature extractor of a neural network is just a bunch of weighted sums followed by a function.

For simplicity reasons the feature extractor of a network is organized in layers.

All output from the previous layer(s) is treated by subsequent layers (in general).

There are different layer types:

fully connected normally used in the final classification layer

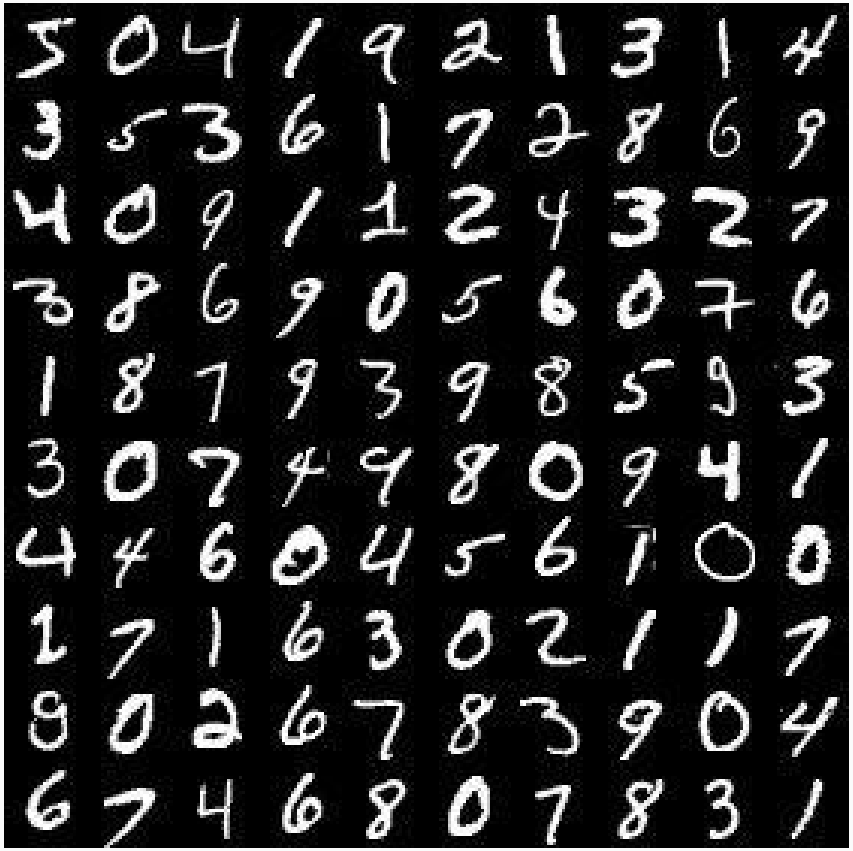
convolutional trainable filterbanks well-suited for image processing. Like lots of small separate fully connected layers.

recurrent designed for sequences. Able to run on arbitrarily sized inputs.

pooling reduce the size of a feature in a non-parametrized fashion.

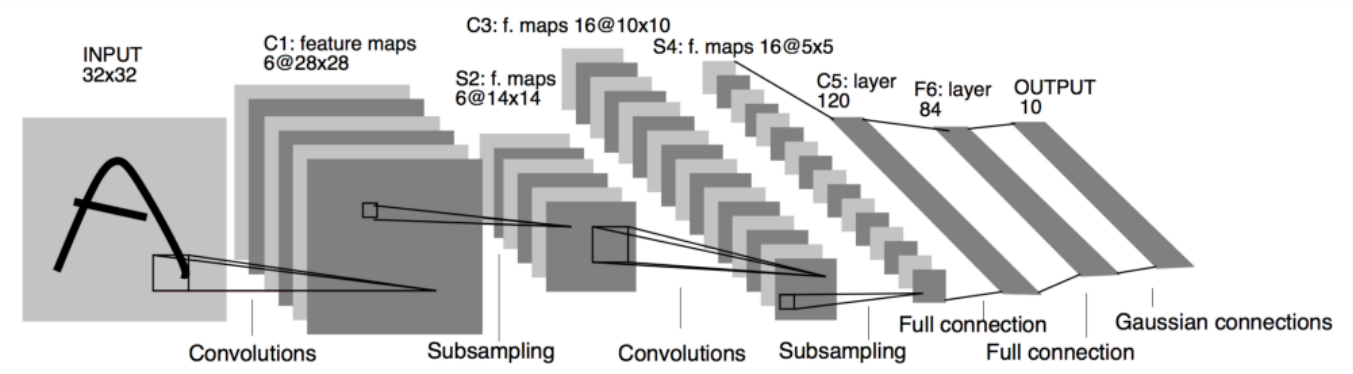
Supervised learning - MNIST

MNIST dataset of 50000
handwritten numerals.
10 classes, 28px $\times$ 28px images



Neural networks - Network architectures

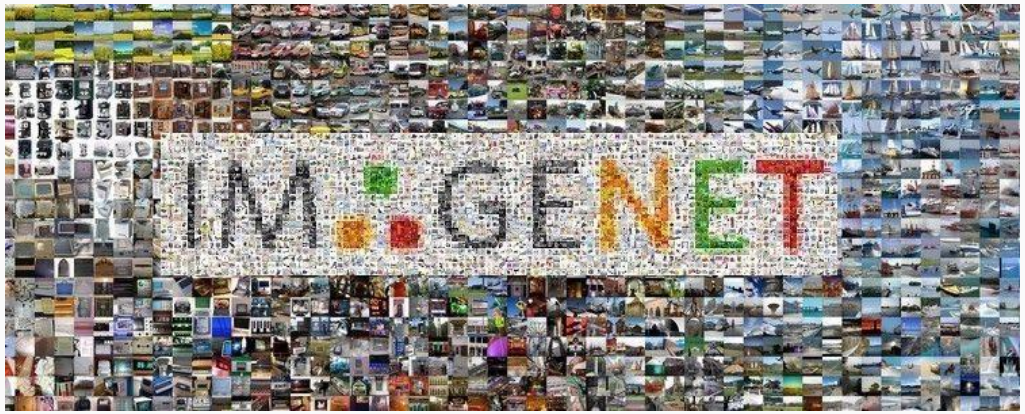
LeNet: 99% accuracy on MNIST classification task (1998)



Supervised learning - ImageNet

Really big image dataset (14 million images, 22k categories)

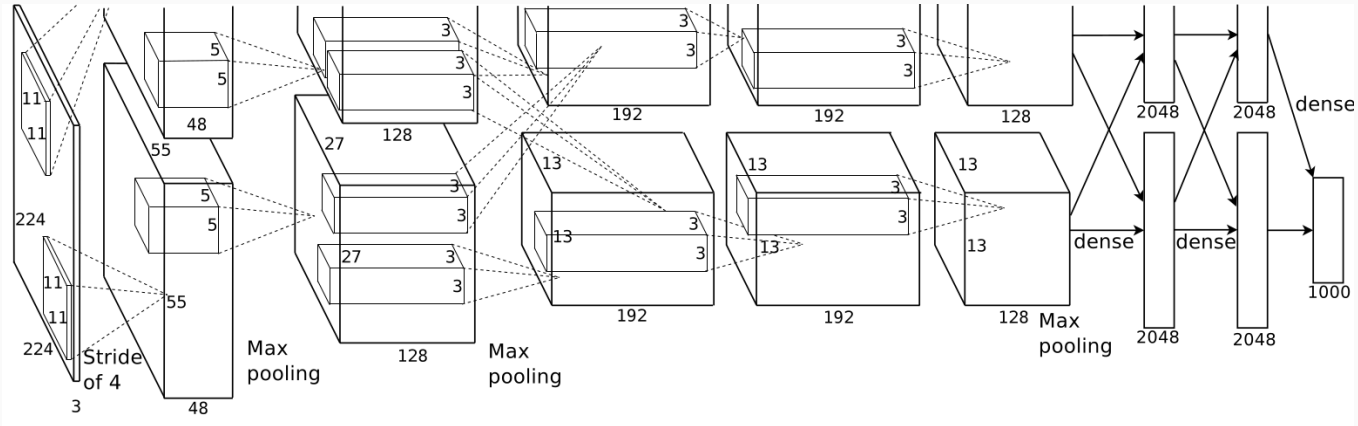
Classification usually done on challenge subset (1.2 million images, 1000 categories)



Neural networks - AlexNet

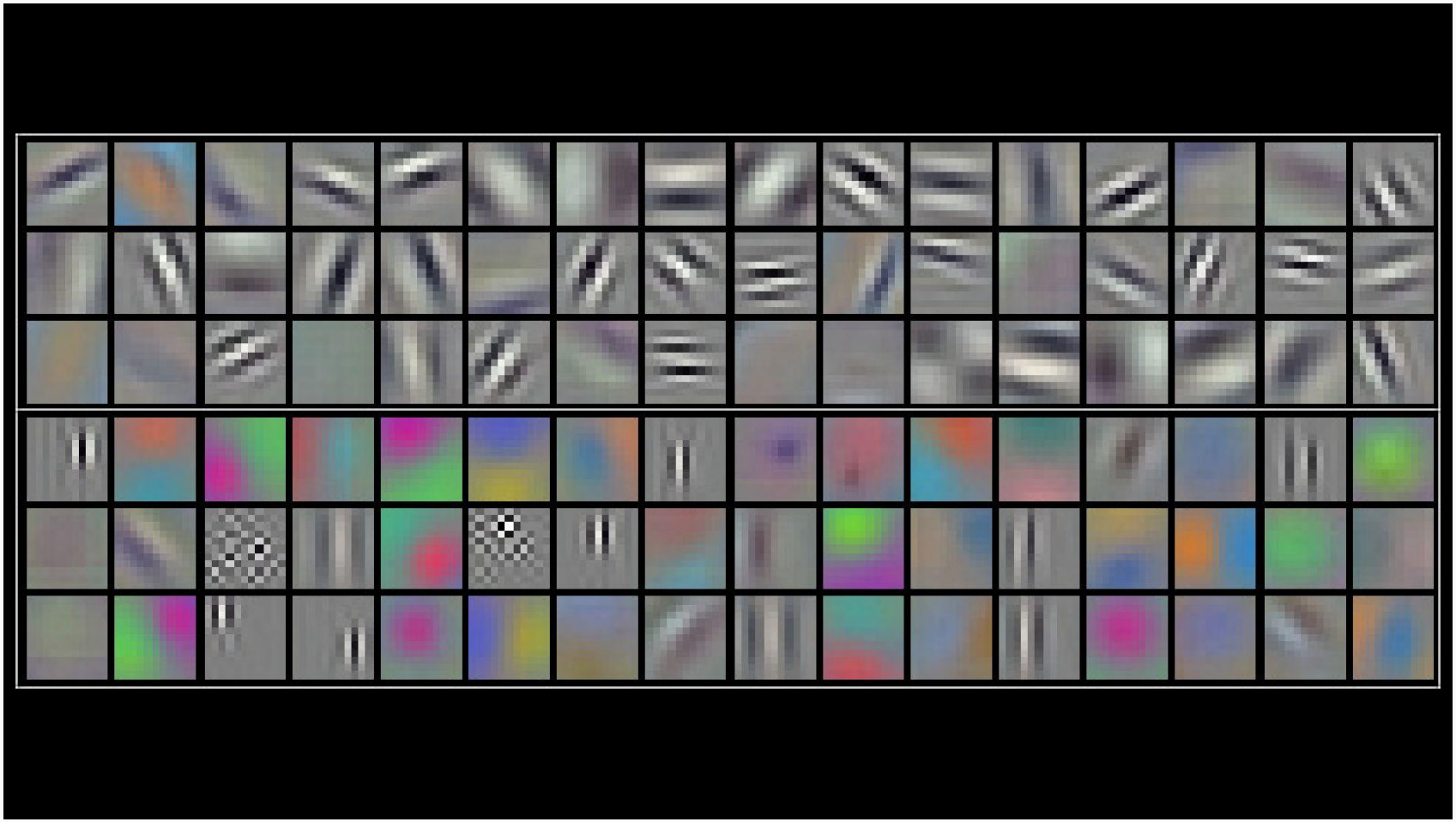
Entry for ImageNet challenge with much higher accuracy than older methods (2012).

First method to run on graphics cards (GPU). Makes everything else possible!



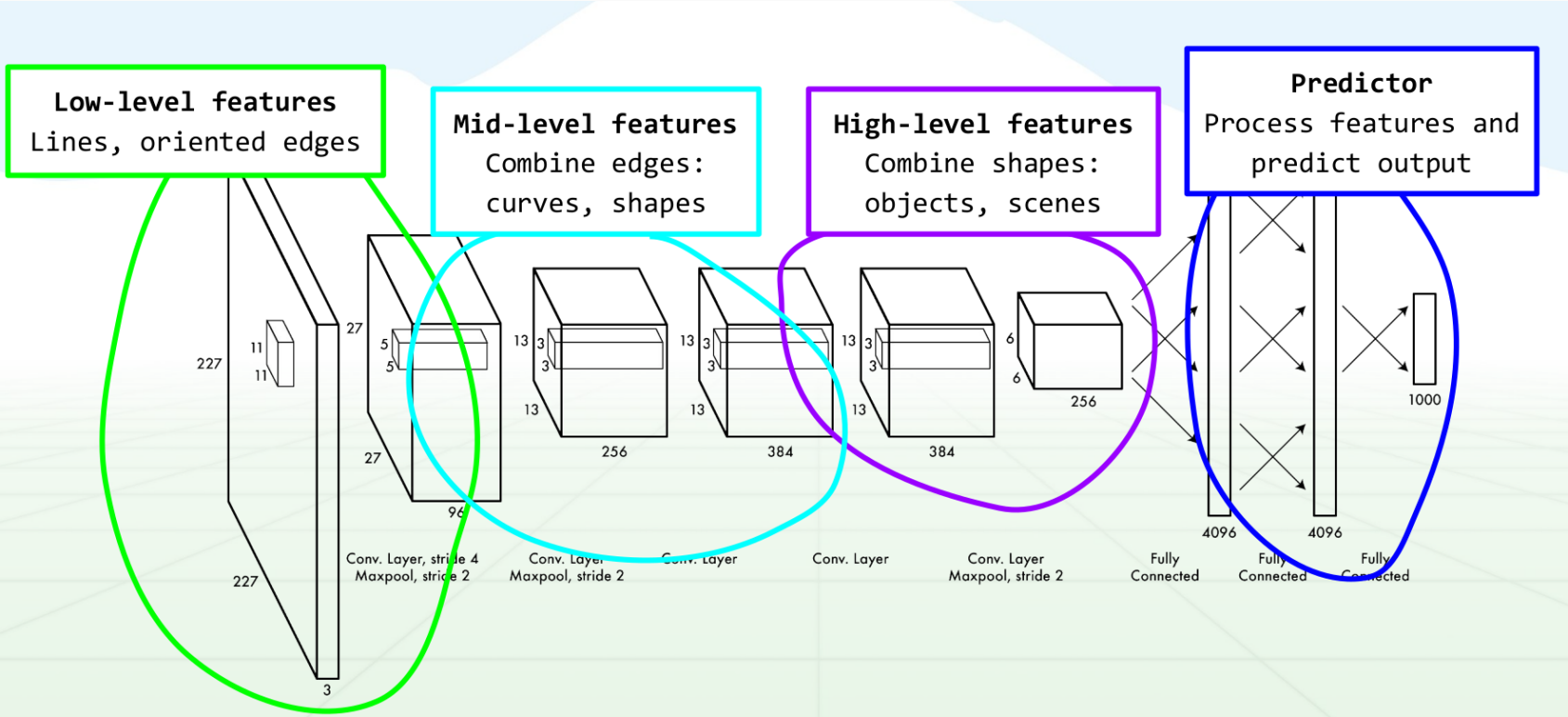
Neural networks - Classification networks

These networks are learning features!



Neural networks - Classification networks

Layers deeper in the network learn higher level features!



Neural networks - Classification networks

There are dozens of other classification networks (VGG, Resnet, GoogLeNet, Inception, ...)

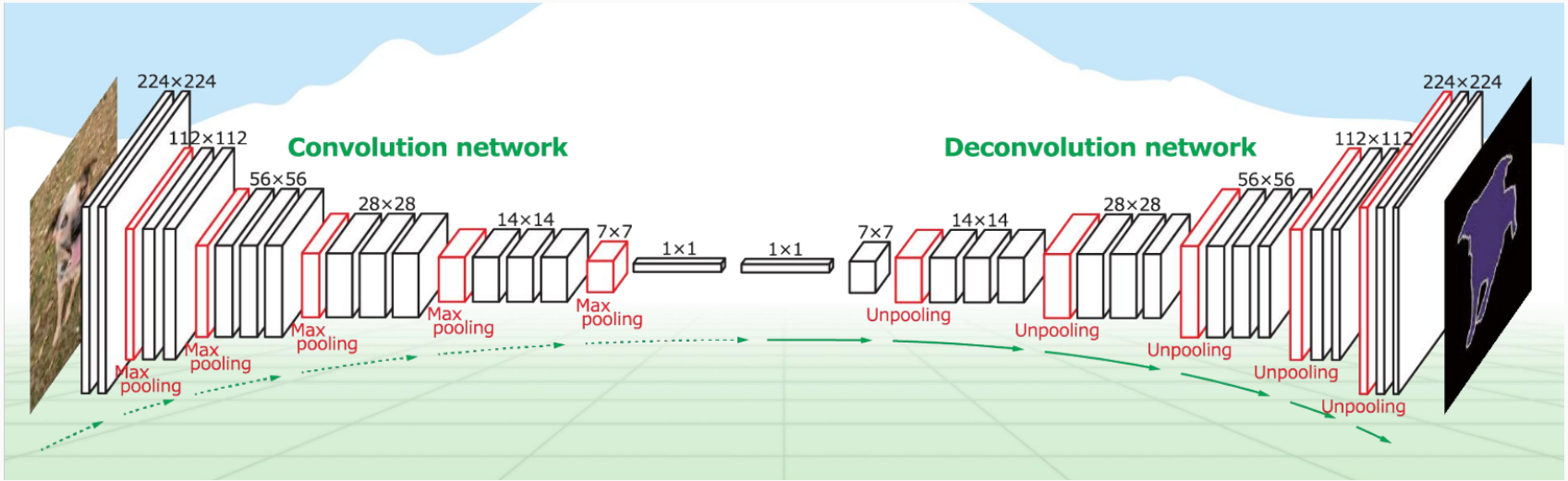
Classification networks work really well.

We can reuse existing the feature extraction part of classification networks for other tasks.

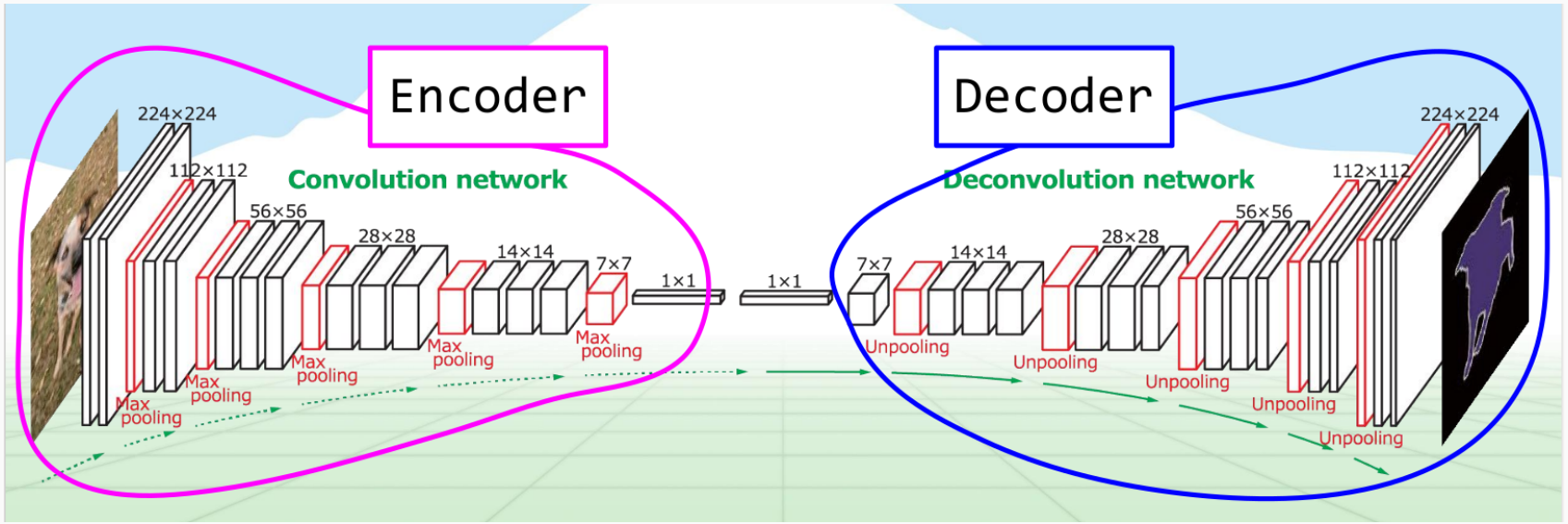
Even for dissimilar domains.

Neural networks - Semantic Segmentation

Neural networks - Semantic Segmentation

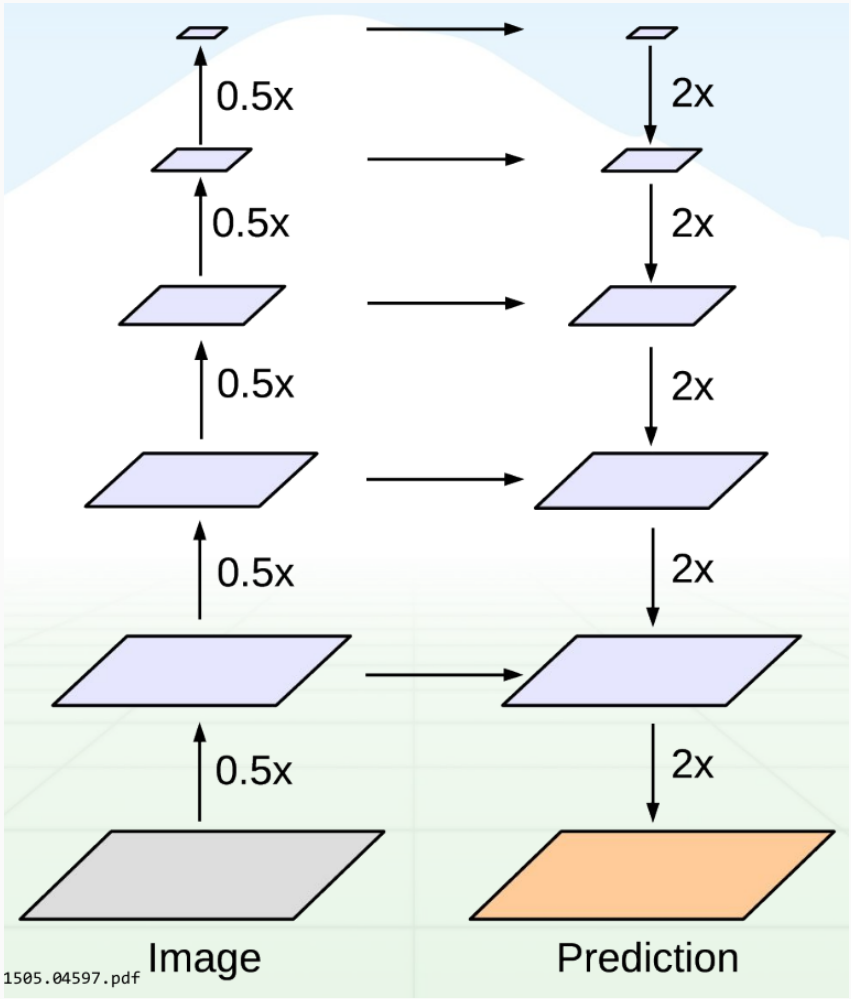


Neural networks - Semantic Segmentation



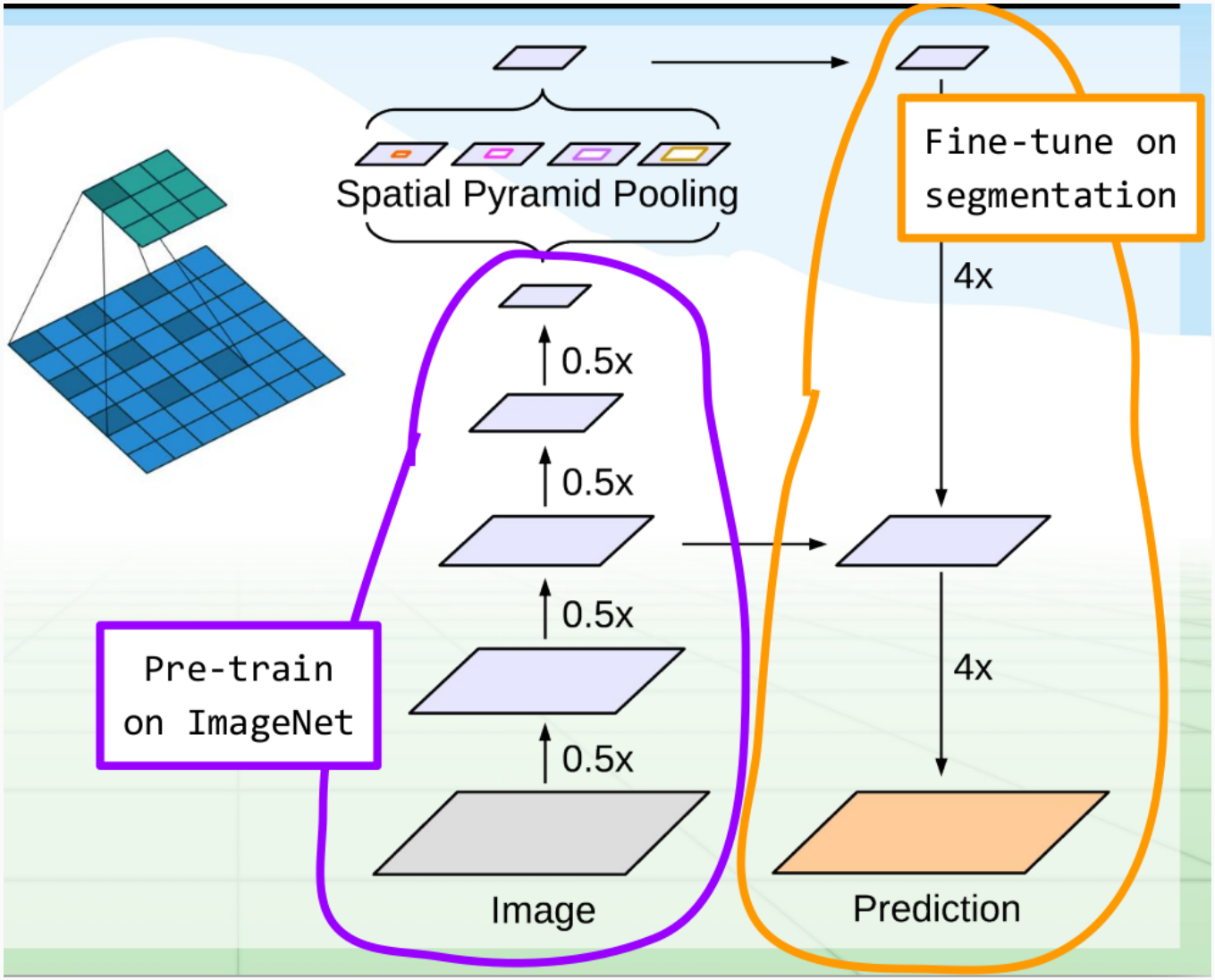
Semantic Segmentation - U-Net

U-net/Segnet improves the resolution by incorporating higher resolution features.

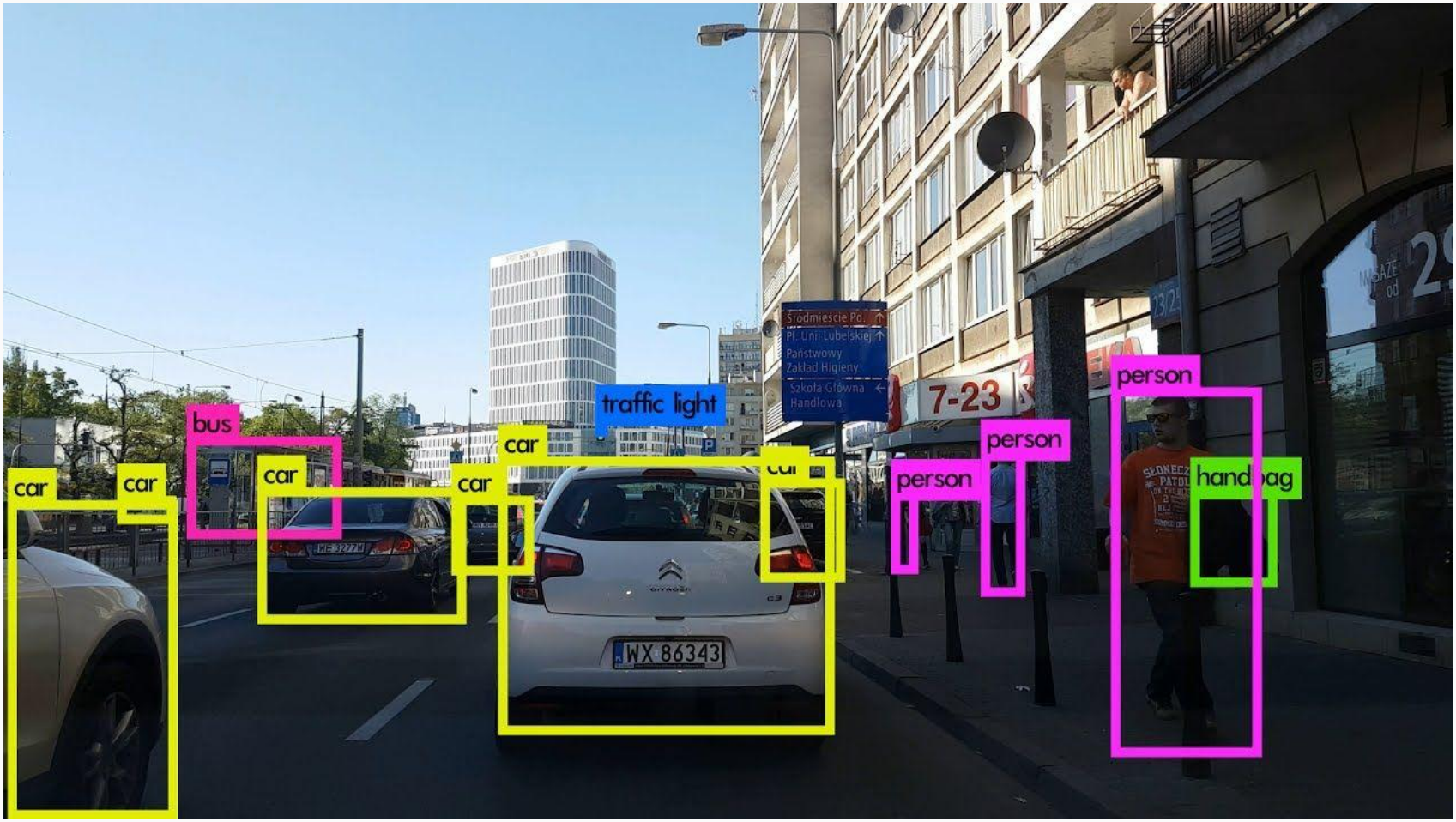


Semantic Segmentation - DeepLabv3

DeepLabv3 uses a specialized type of pooling and convolutions to do the same thing as U-Net.

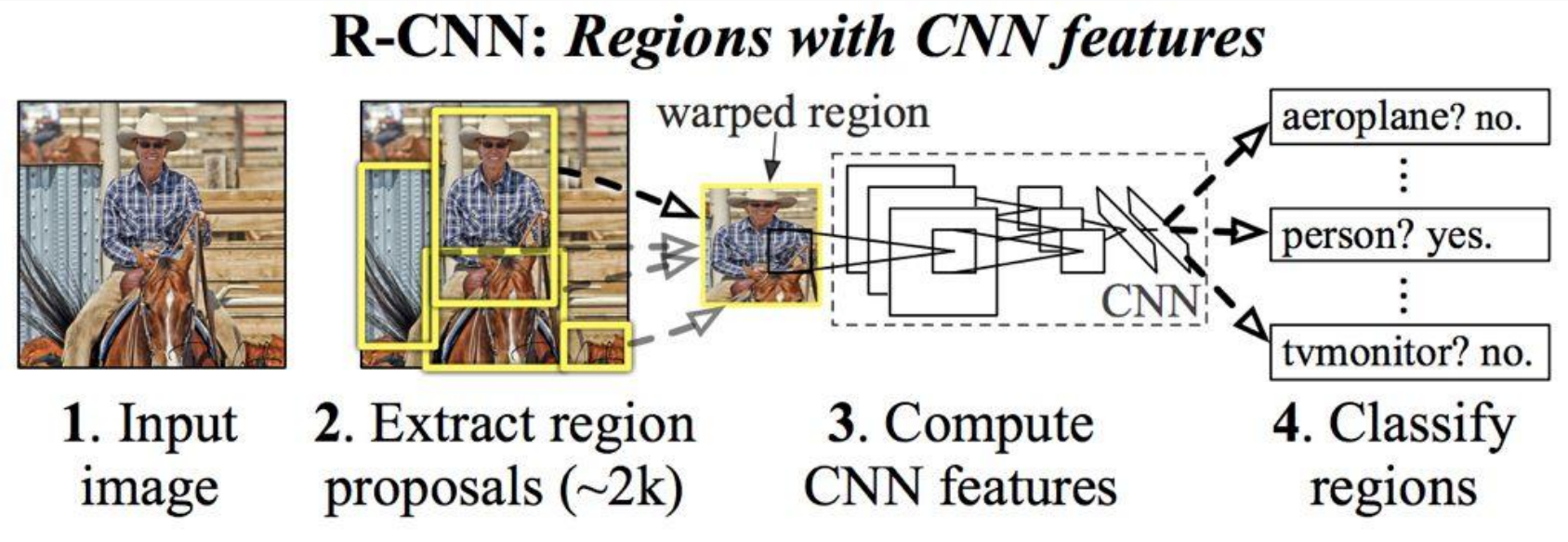


Neural networks - Object Detection



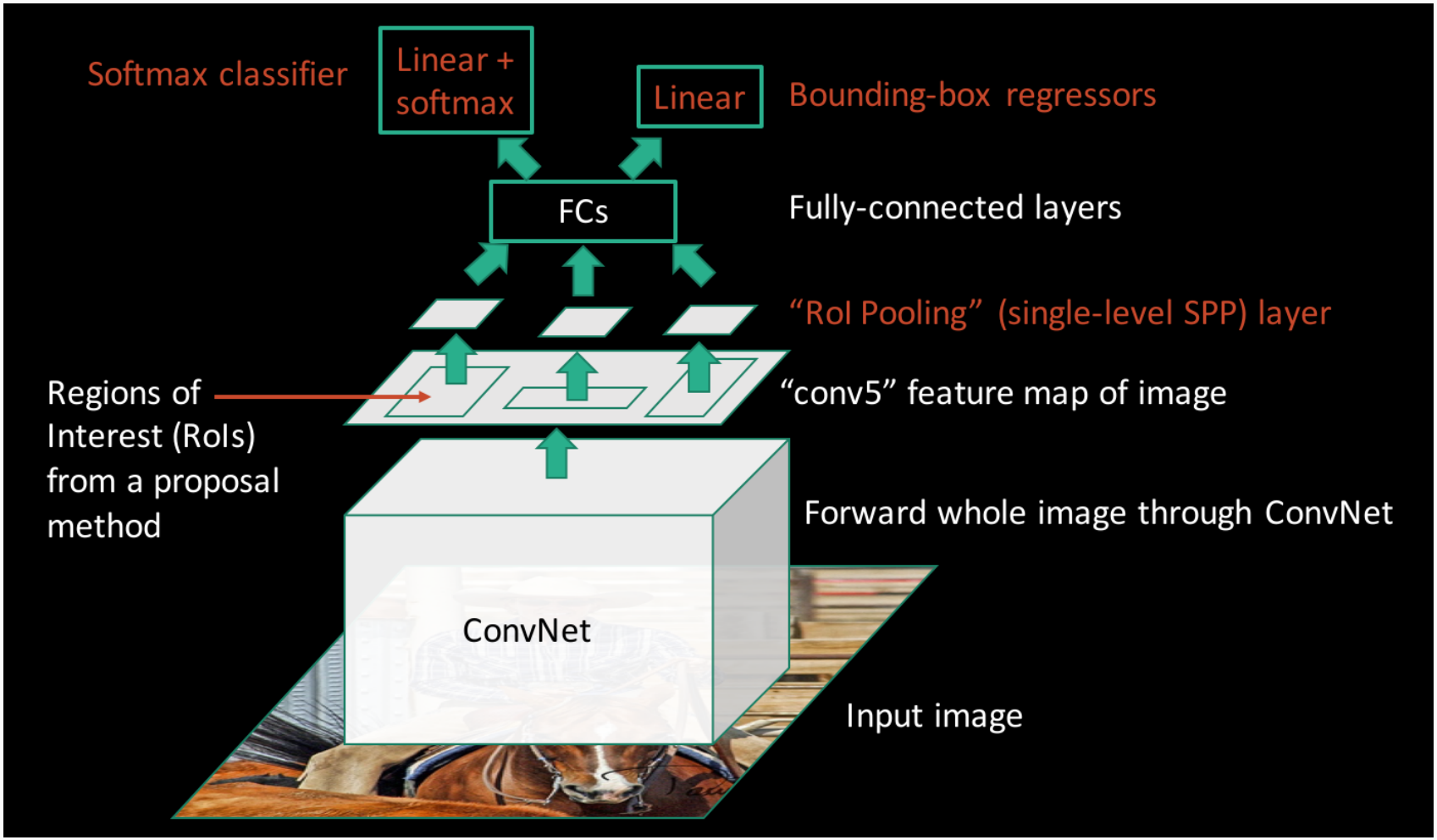
Object Detection - R-CNN

Very, very slow (20s/img)



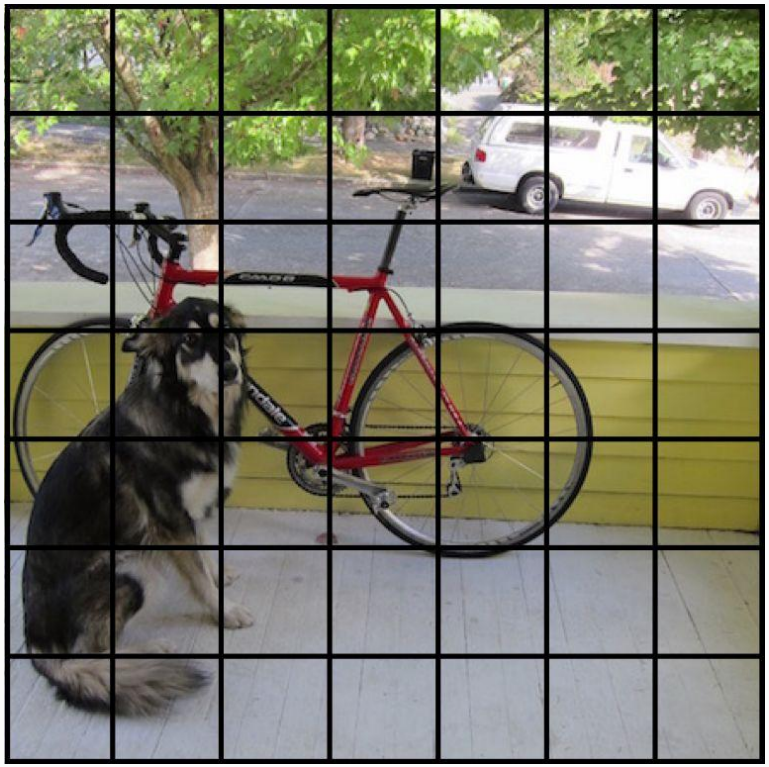
Object Detection - Faster R-CNN

Just slow (2s/img)



Object Detection - YOLO

Split image into a grid + detect boxes and class probabilities at each grid location.



Object Detection - YOLO

Split image into a grid + detect boxes and class probabilities at each grid location.

Multiple versions exist and all are really fast with decent accuracy.

Can be hard to train. Lots of parameters that have to be changed for other datasets.

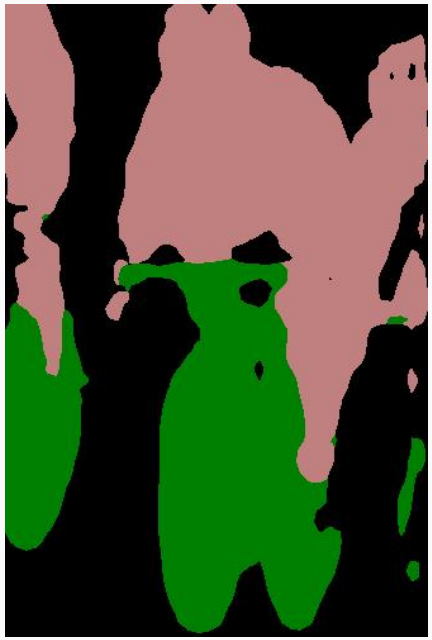
Segmentation vs Detection

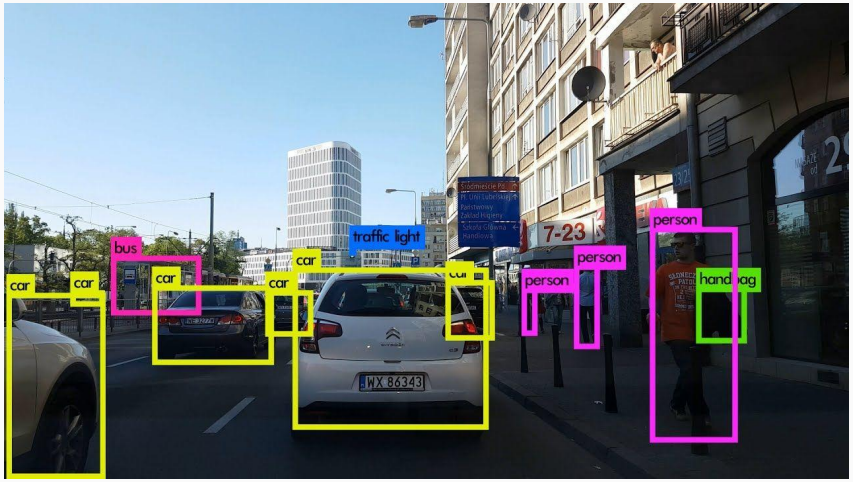
Segmentation:

- Pixel-level labels
- Category only

Detection:

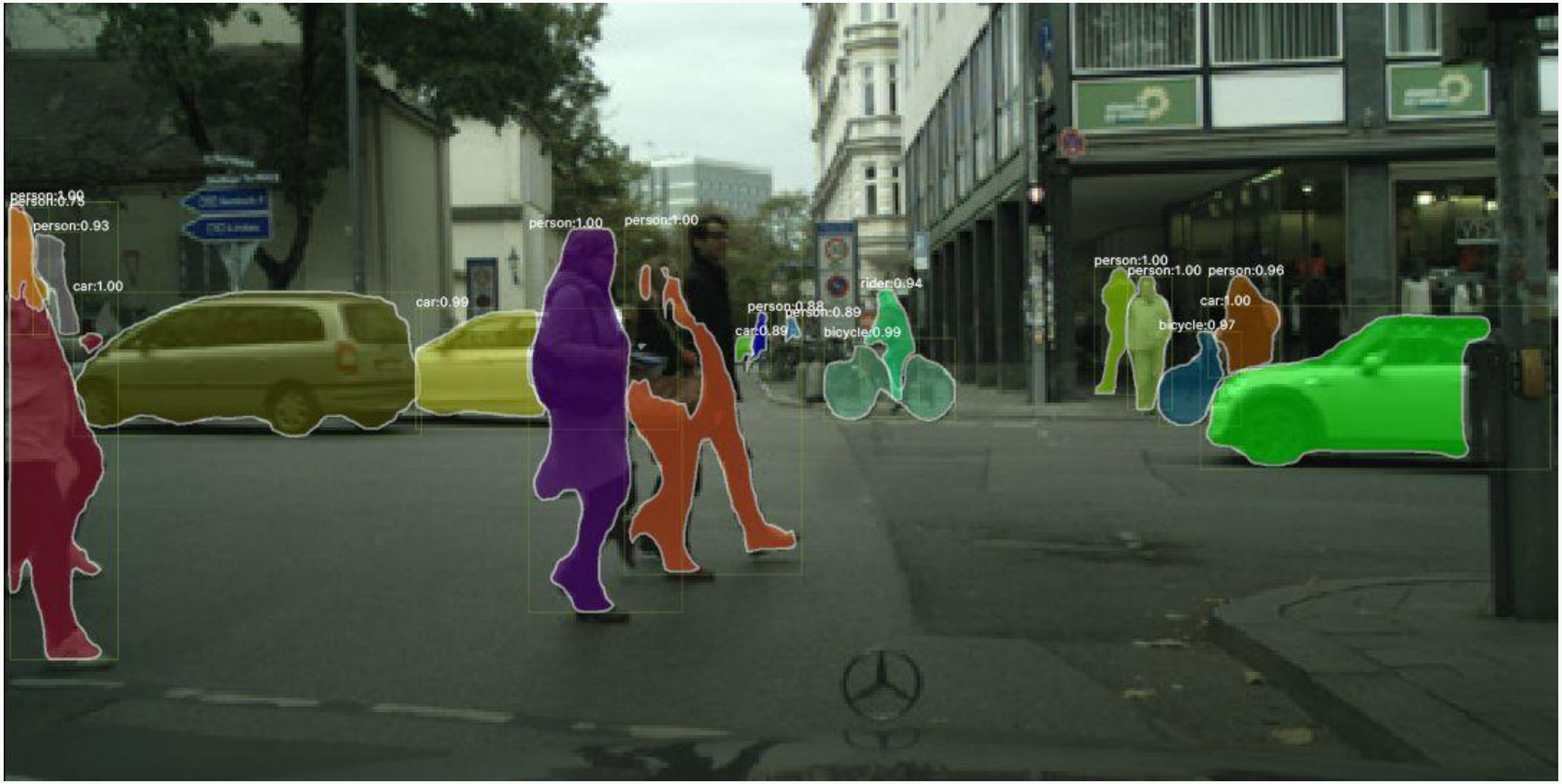
- Bounding box labels
- Category + instance





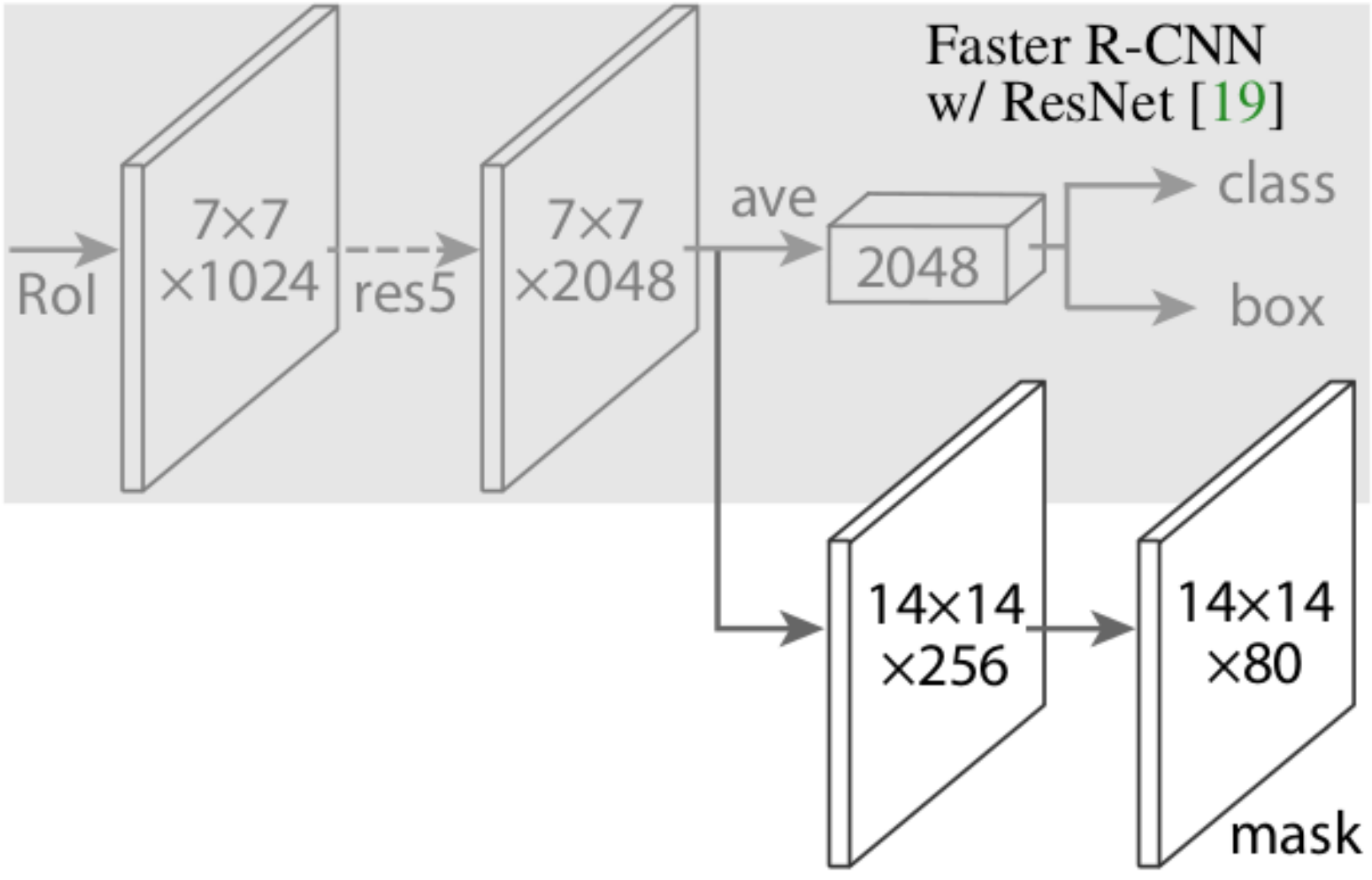
Instance Segmentation

Given an image produce instance-level segmentation: Which class does each pixel belong to and to which instance?



Instance segmentation - Mask R-CNN

Similar to R-CNN (sloooow) but predict mask instead of box.



Instance segmentation - SOLO

Jointly detects instances and labels masks.

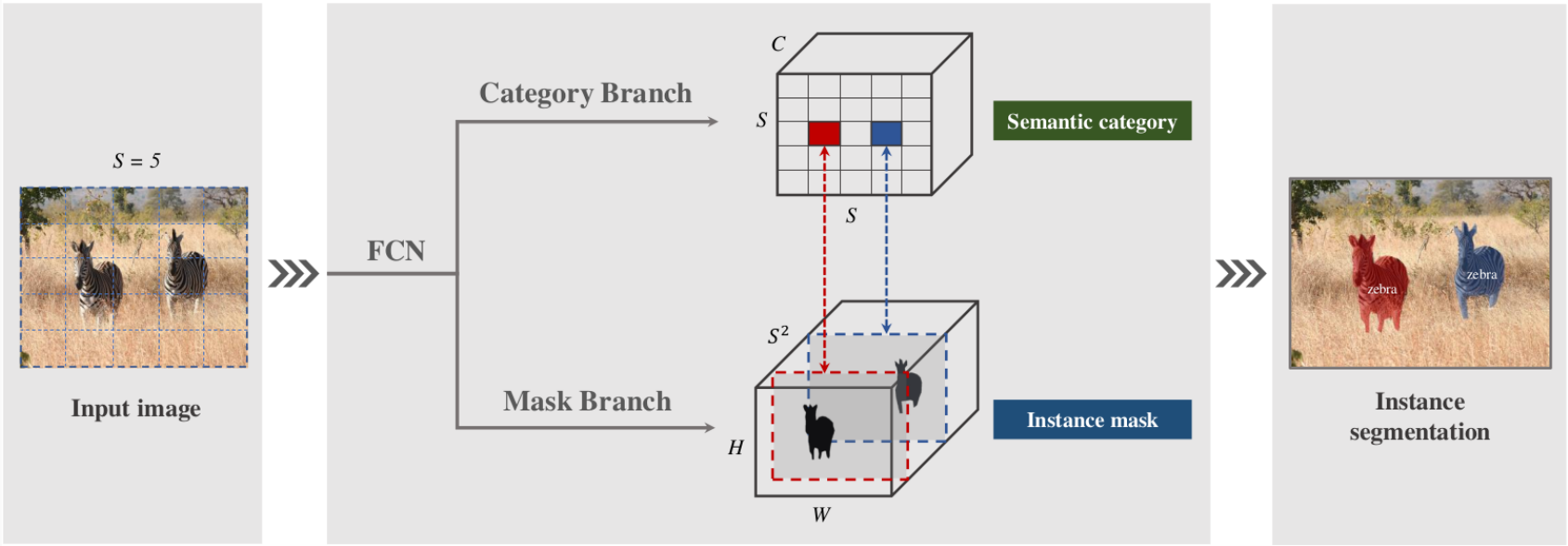


Image captioning

Given an image: Extract features (we know how to do this!) and generate sequences (e.g. with a recurrent network).

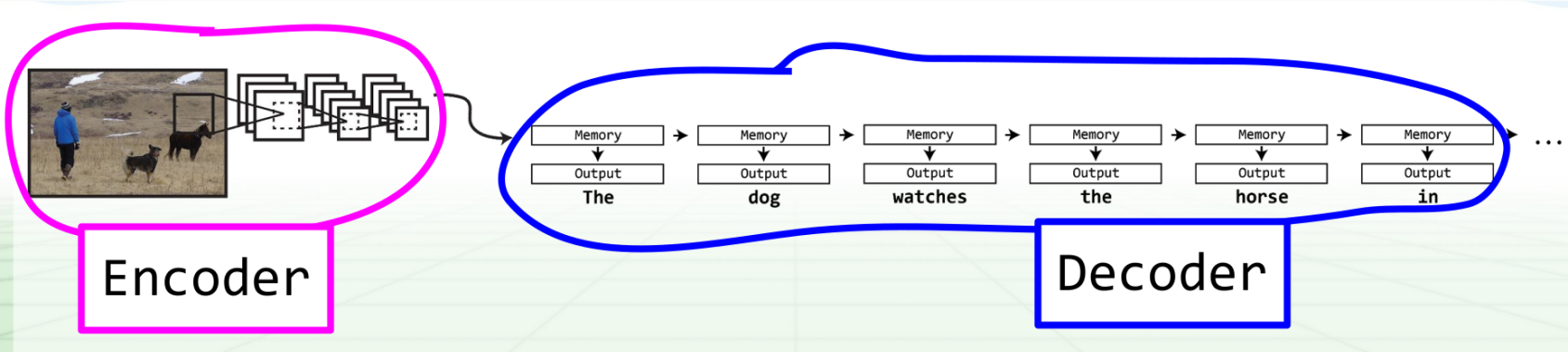


Image captioning - The good

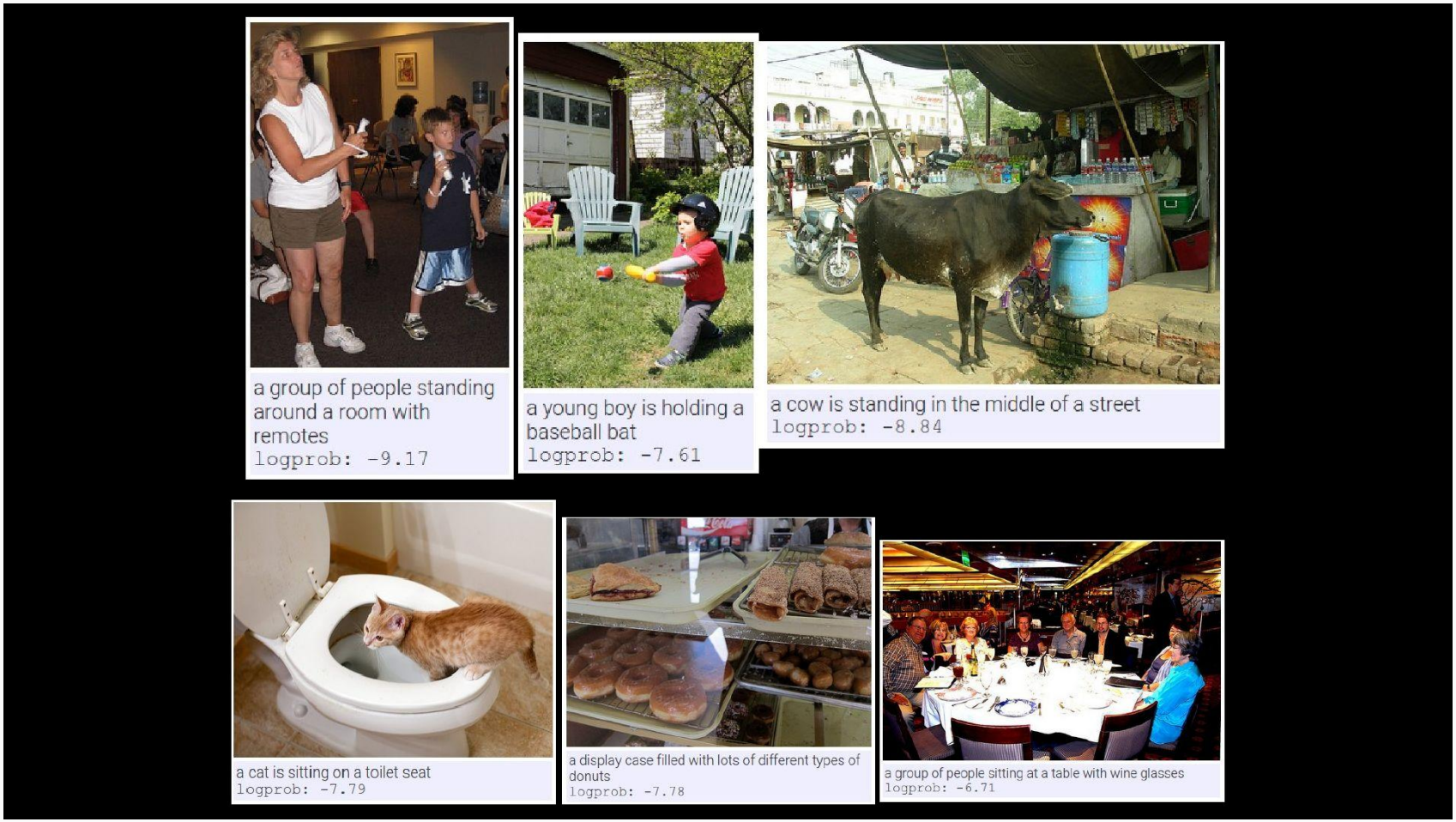


Image captioning - The Bad

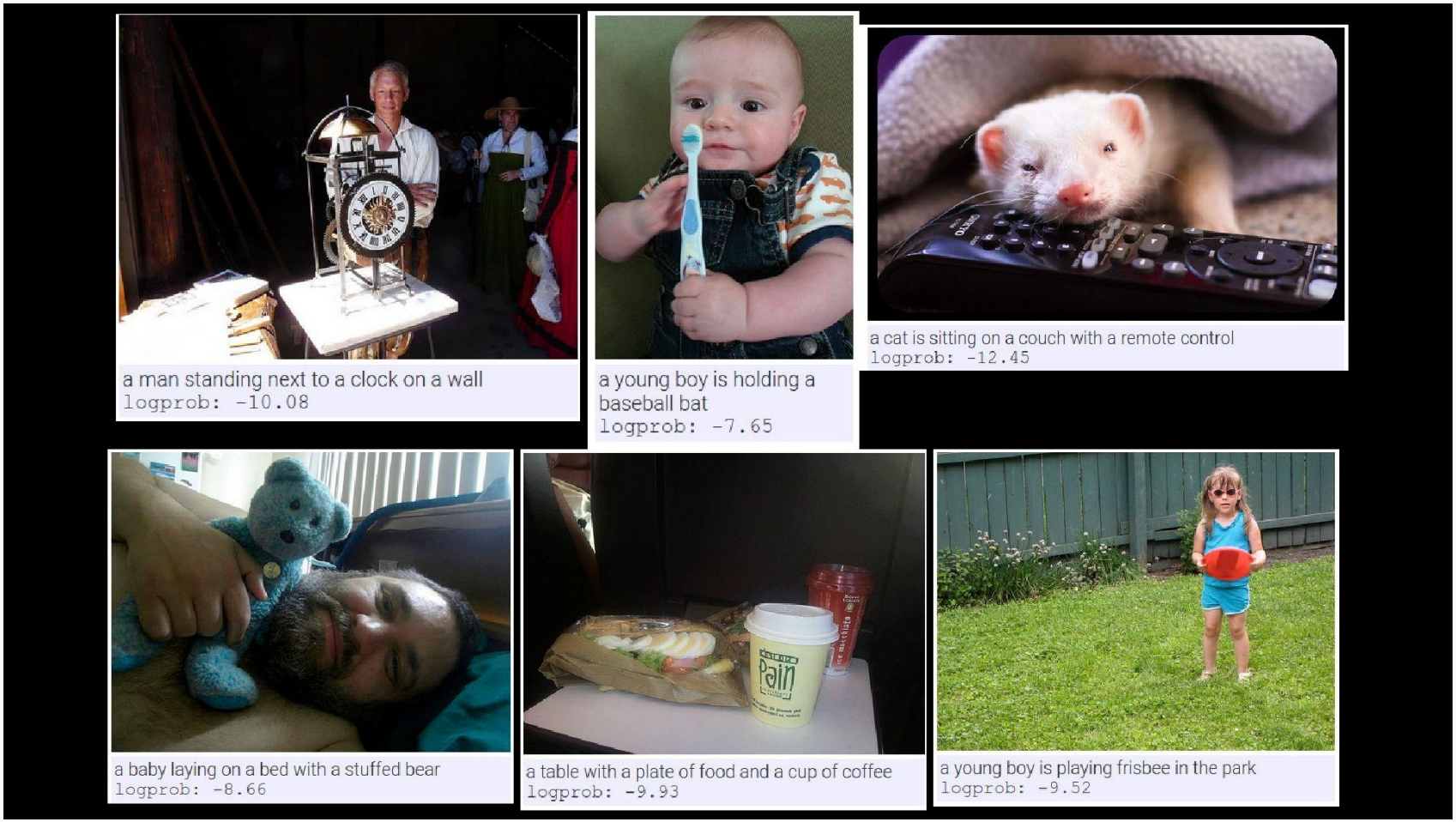


Image captioning

Works OK-ish by now.

Datasets are really large (millions of samples).

Language is a lot more difficult than images. Features don't transfer to other use cases (for reasonably sized models).

Most recent research can't be replicated or adapted without silly amount of ressources.