

# Object Recognition

## DD2423 Image Analysis and Computer Vision

Mårten Björkman

Robotics, Perception and Learning Division  
School of Electrical Engineering and Computer Science

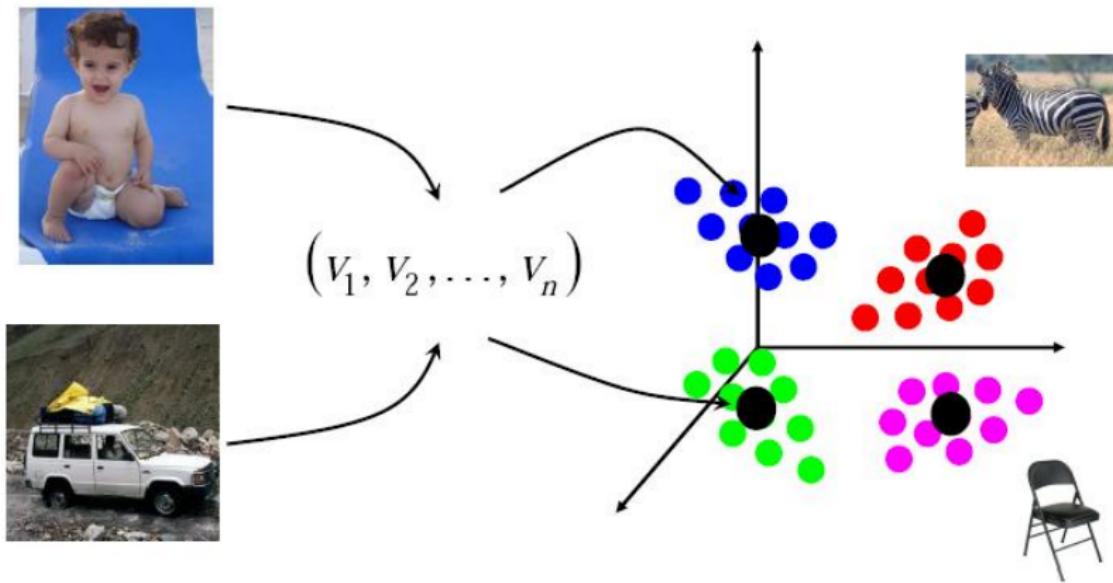
December 6, 2021

- Recognition - Is this my cup?
  - Only requires a model of one cup.
  - What makes the individual cup unique?
  - Classical methods work very well.
- Classification - What is this?
  - Requires a model that captures all cups.
  - How to group objects into classes?
  - Deep learning works very well.
- Detection - If there is a cup, where is it?
  - A much harder problem.

# Common for all methods

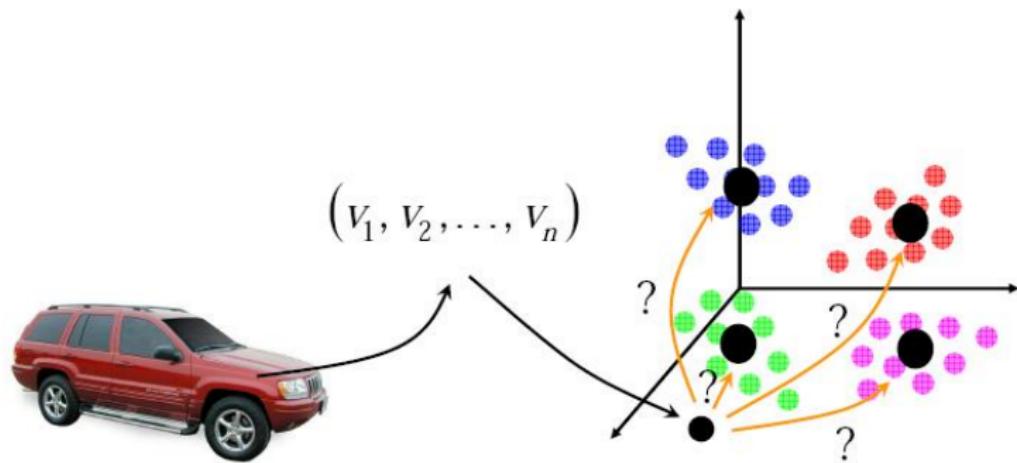
- Two or more hypotheses.
  - Is this a cup or not?
  - Is this a book, cup, monkey, pen or ...?
- Need for representation.
  - As small as possible, otherwise redundant.
  - A database of objects can be very large.
- Need for metric, such that representations of
  - ... the same class are close.
  - ... different classes are distant.

# Recognition (Step 1: representation)



Take an image (region) and represent it in some feature space.

## Recognition (Step 2: classification)



Given a new image in feature space, determine class it belongs to.

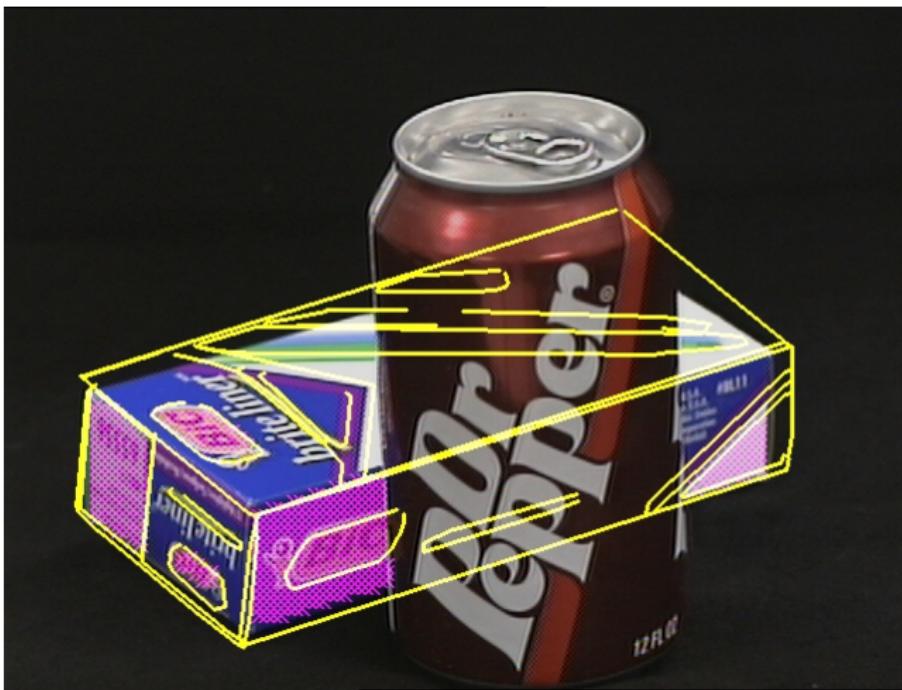
- Supervised vs unsupervised
  - Supervised: annotated training examples exist
  - Unsupervised (clustering): no annotated training examples
- Generative vs discriminative
  - Generative: models how things look like
    - ex. face = 2 eyes + 1 nose + 1 mouth
  - Discriminative: models the difference between things
    - ex. face vs non-face

- Appearance of object may vary due to different:
  - Lighting conditions (shadows, colours)
  - Poses (viewing directions)
  - Deformations (changes in shape)
  - Clutter (occlusions)
  - Projective sizes (distances)
  - Cameras (projections, distortions)
- Thus... matching has to be (more or less) invariant to these.

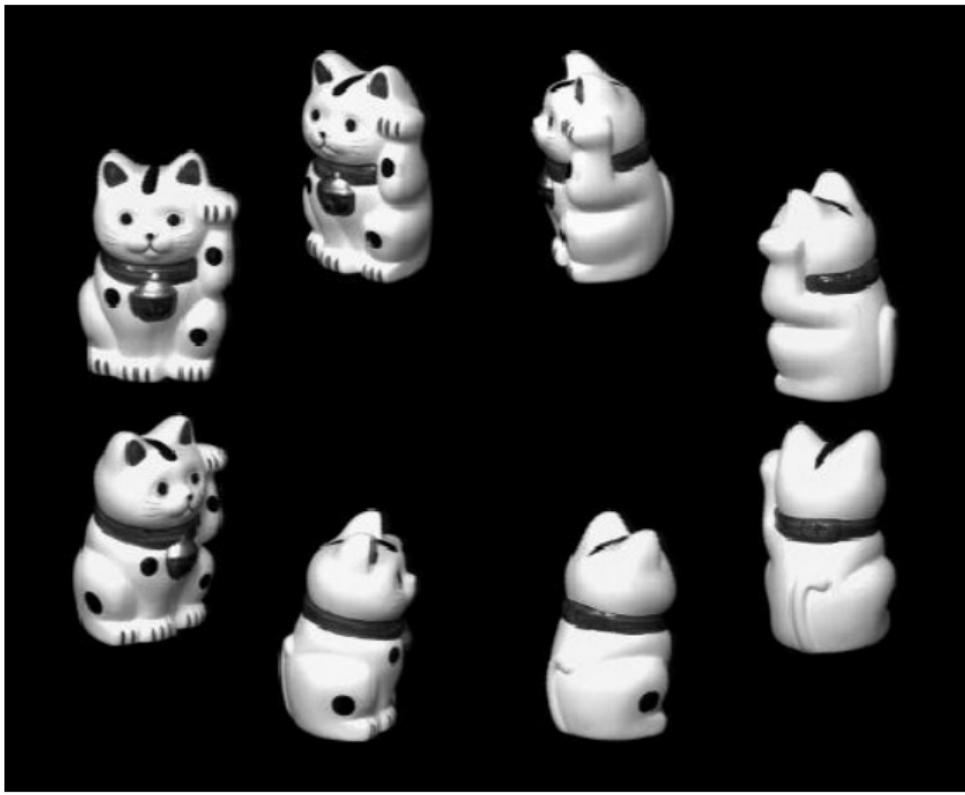
# Illumination Invariance



# Occlusion



# View-point Invariance

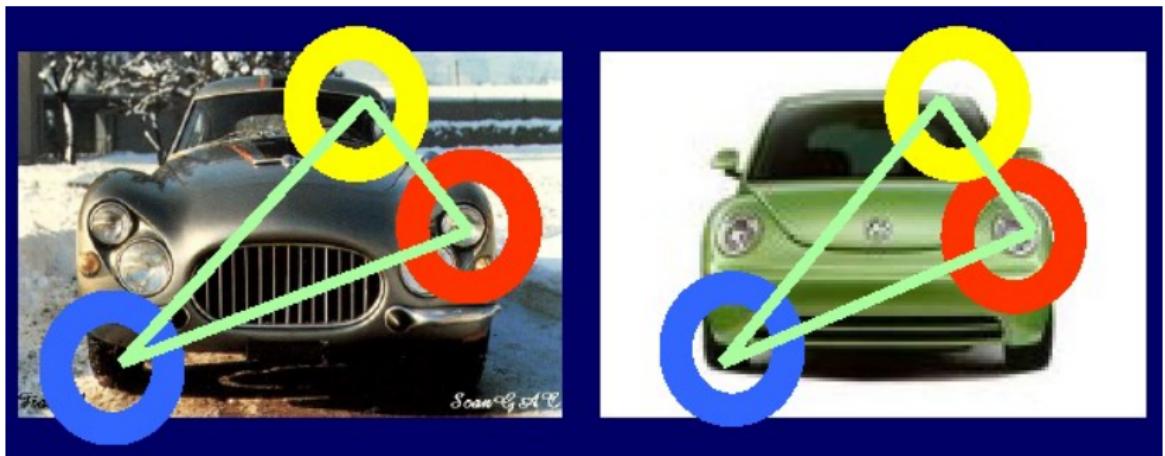


- Template based (dense)
  - Store some “image” of object (characters, faces).
  - Relative positions very important.
  - Normalized (lighting, scales) for invariance.
- Feature based (sparse)
  - Store sets of characteristic features (corners, contours).
  - Relative positions flexible.
  - Only keeps information that can be made invariant.
- Statistics based (usually dense)
  - Store histograms of image data (edges, colours).
  - Positions are disregarded (textures).
  - Each kind of data matched invariantly.

Methods typically contain:

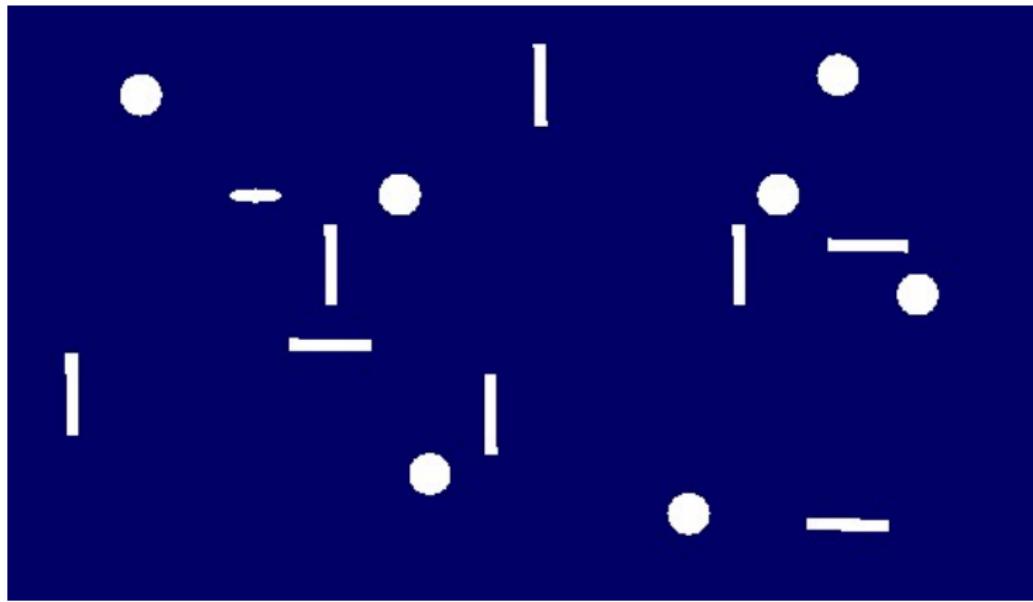
- Some feature detector
  - High curvature points (corners)
  - Scale-space blobs
- Some invariant descriptor(s)
  - Templates (small windows around features)
  - Statistics (edges, colours, etc.)
  - Combinations of templates and statistics
- Some way of combining features
  - Voting or similar combination measure
  - Often using relative position statistics

# Recognition by parts



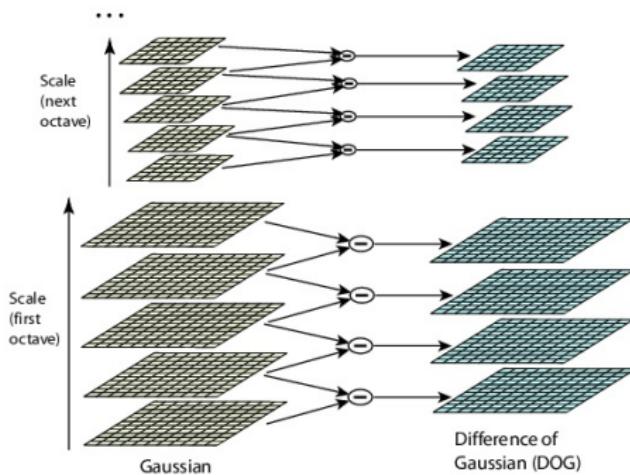
Each part might look like many other parts, but combined they might strengthen a hypothesis.

# Importance of feature grouping



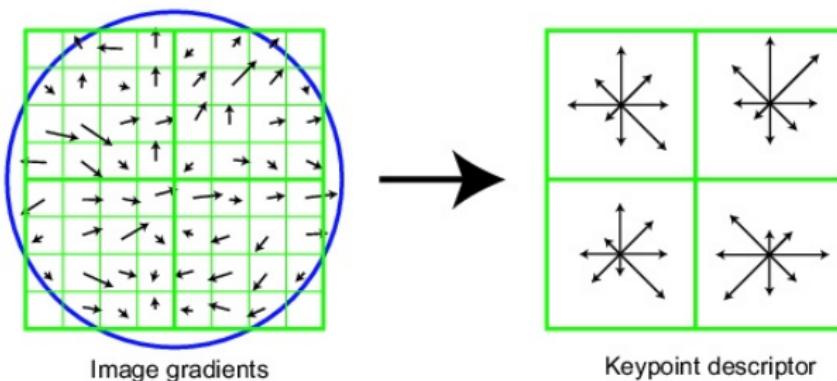
Can you see the face?

# SIFT features (detector)



- Detected as peaks in Differences of Gaussians.
  - Approximation of Laplacian (blob detector).
- Multiple scales allow features to be detected regardless of scales.

# SIFT features (descriptor)



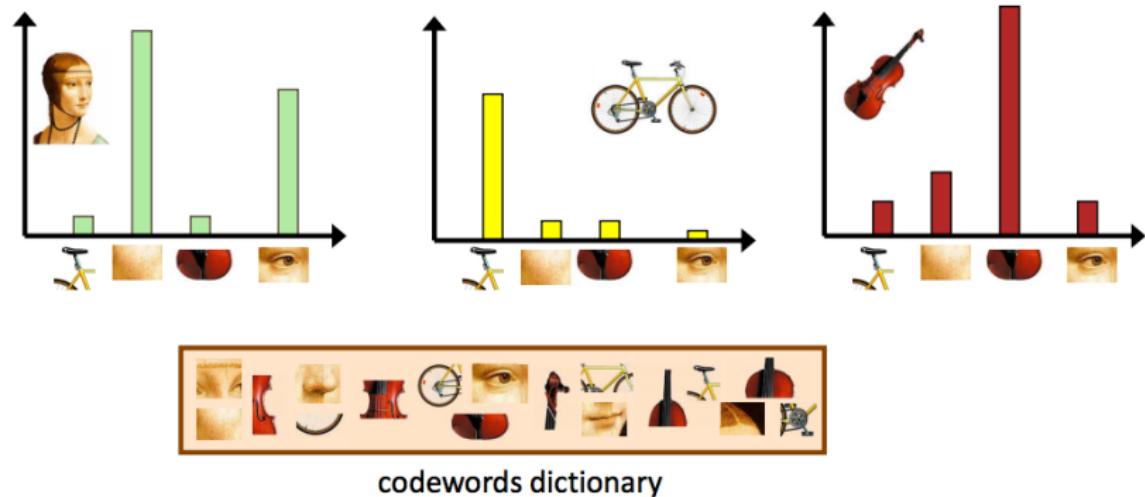
- Descriptors consist of local gradient direction histograms.
- Dominating direction as reference  $\Rightarrow$  rotation invariance.
- Window size from scale-space detector  $\Rightarrow$  scale invariance.
- Normalize feature vector  $v$  to  $|v| = 1 \Rightarrow$  luminance invariance.

# SIFT features (matching)



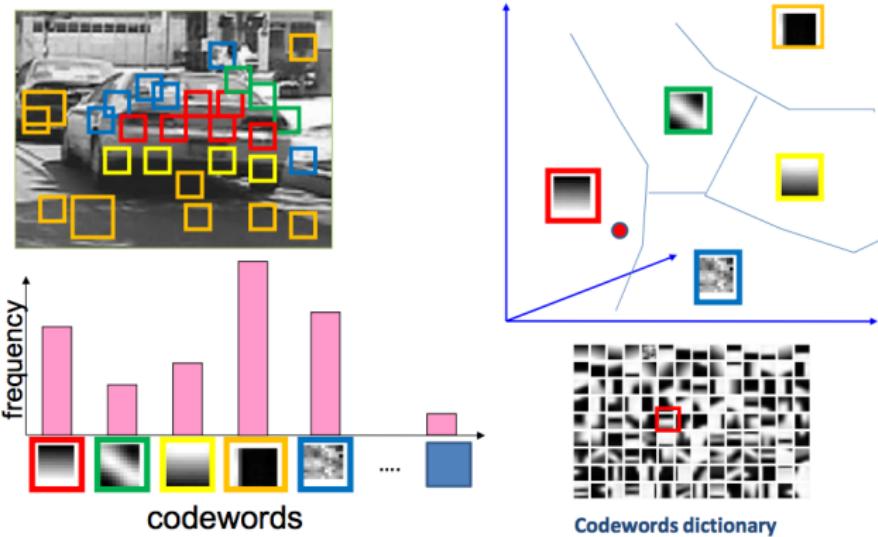
- Match features between image and model.
- Fit a model to remove mismatches (e.g. homography).
- Is there a frog or locomotive in the scene?
  - If enough features are matched the question is Yes.
- Still a good method to confirm the identity of a detected object.

# Bag of Words (BoW): How to go to millions of images?



- Idea: represent images as a histograms of features (often SIFT).
- Google: a document is represented as a histogram of words.

# Bag of Words (BoW): Summary of steps

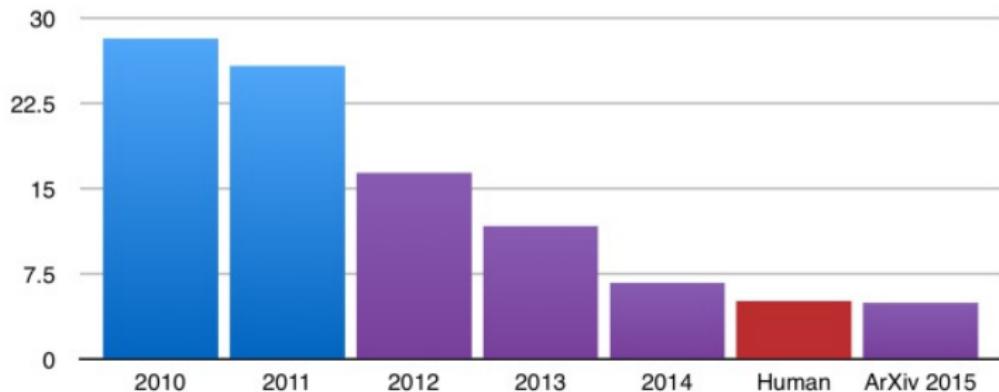


- Step 1: Feature extraction (e.g. SIFT)
- Step 2: Clustering in feature space (e.g. K-means)
- Step 3: Collect histograms of feature numbers
- Step 4: Classify histograms into different object classes

# Deep Networks for object recognition

- Neural networks were long forgotten in computer vision.
- Recently, deep neural networks have become state-of-the-art.

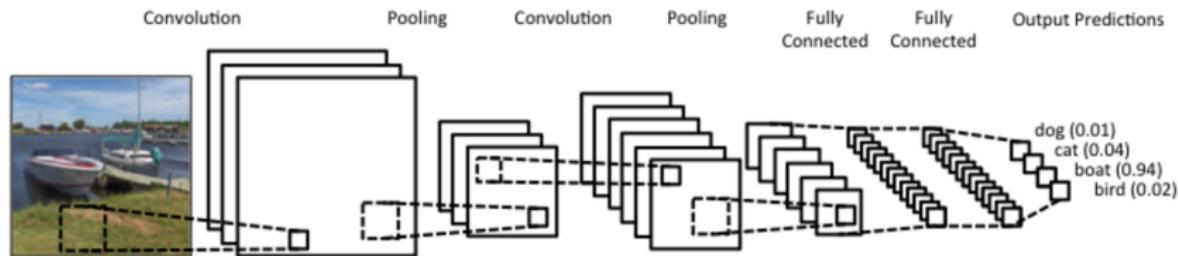
ILSVRC top-5 error on ImageNet



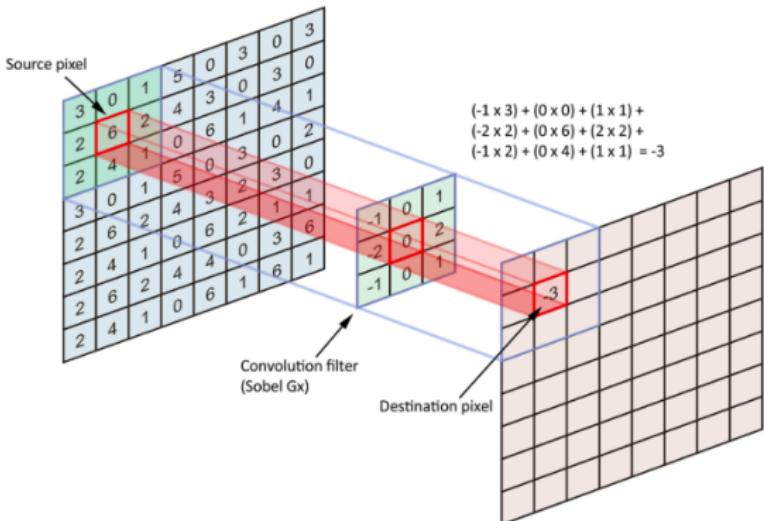
The best methods today have a top-5 error of about 1%.

# Repetition: Convolutional Neural Networks (CNN)

- Idea: have a very simple structure, but learn everything.
- Each layer includes four steps:
  - Convolutions (normal filtering operations)
  - Non-linear operator (e.g. ReLU: set negative values to zero)
  - Pooling (e.g. find local maximum and subsample)
- Last layers are fully connected.

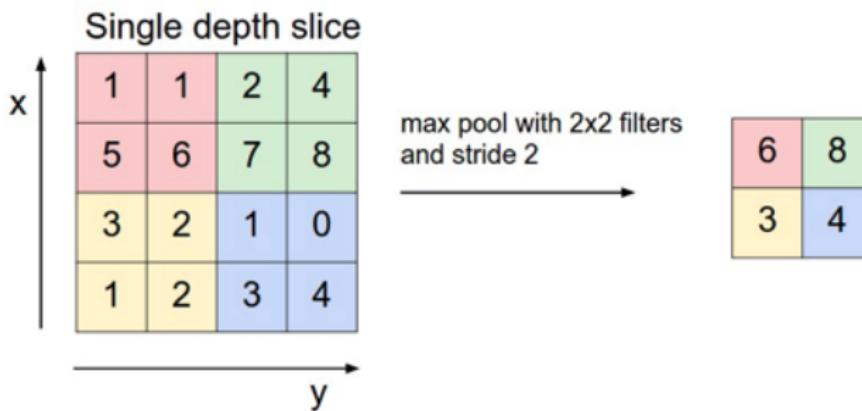


# CNN: convolutional layers



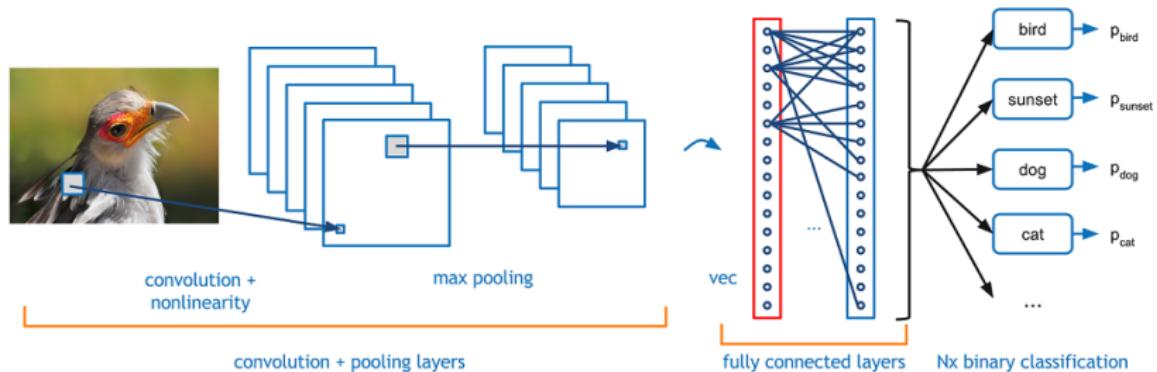
- Similar to the image filters we have seen before.
- However, filters are learned from large amounts of data, and kernels are typically in 3D over multiple input channels.
- Followed by non-linear activation function.

# CNN: pooling layers



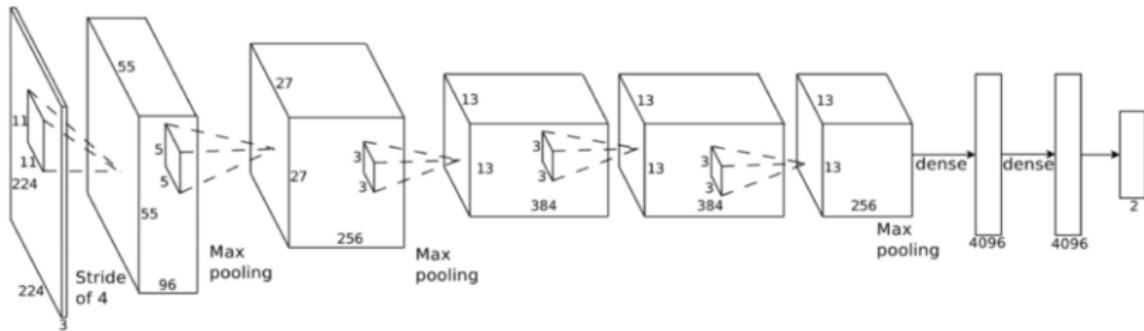
- Pooling reduces image sizes by local maximisation or averaging.
- Gradually makes result more translationally invariant.

# CNN: fully-connected layers



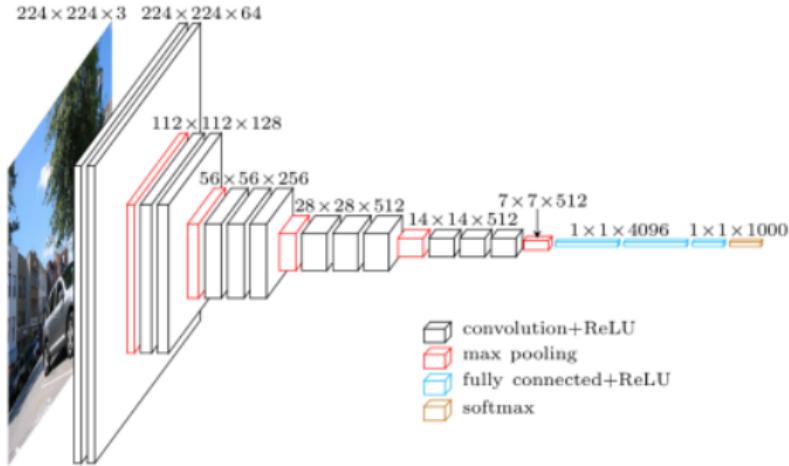
- After flattening data, last layers are fully connected.
- It is where the actual classification is done.
- Feature learning: Train for classification, but use results just before the fully-connected layers for something else.

# AlexNet (2012)



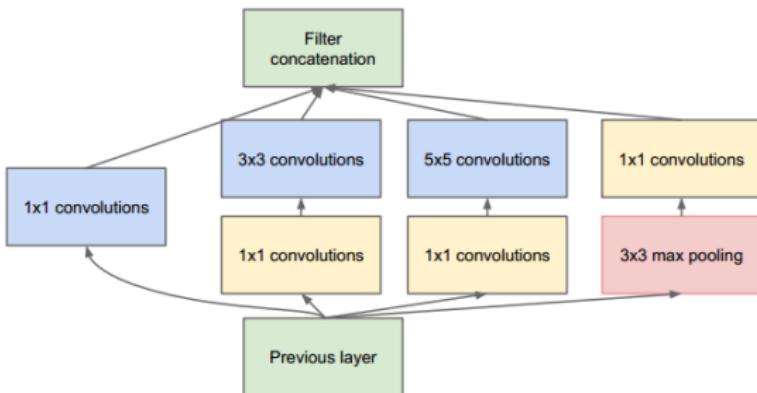
- Based on LeNet (LeCun et al, 1998), but revolutionised image classification.
- Break-through: used ReLU for activation and normalization of activation to allow training of deeper networks.

Krizhevsky et al., "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012.



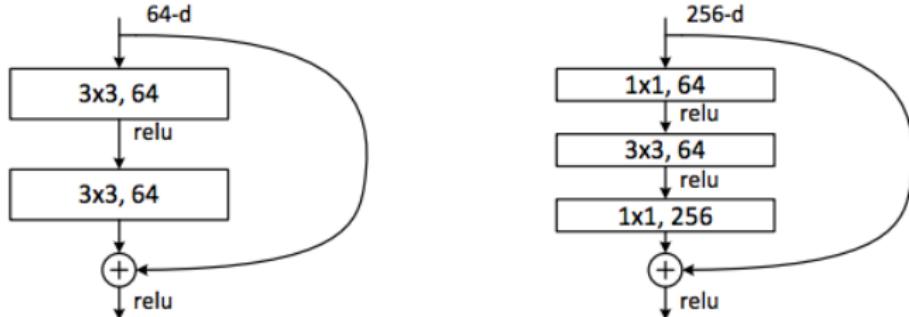
- Novelty: only applied small  $3 \times 3$  filters, but more layers.
- This is better than fewer layers with larger kernels.
- Often used for feature learning, without training your own network.

Simonyan and Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, arXiv 2014.



- Novelty: Used sequences of so called Inception modules.
- 1x1 convolutions are over input channels only, instead of 3D kernels over image space and channels.
- Leads to much fewer parameters to train → faster.

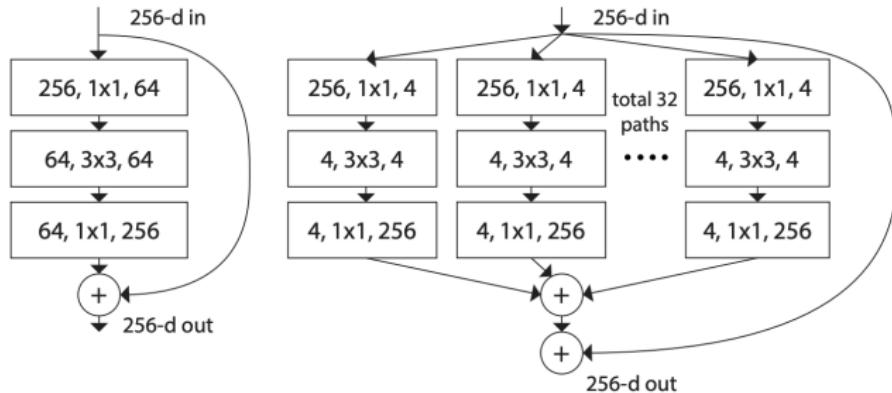
Szegedy et al., "Going Deeper with Convolutions", CVPR 2015.



- Novelty: Used sequences of so-called Residual blocks.
- Faster propagation of errors during training → deeper networks.

He et al., "Deep Residual Learning for Image Recognition", CVPR 2016.

# ResNeXt (2017)



- Novelty: Project down to  $4 \times 32$  dimensions instead of 64
- Same number of parameters, but more ability to learn

Xie et al. "Aggregated Res. Transformations for Deep Neural Networks.", CVPR 2017.

# Results on ImageNet

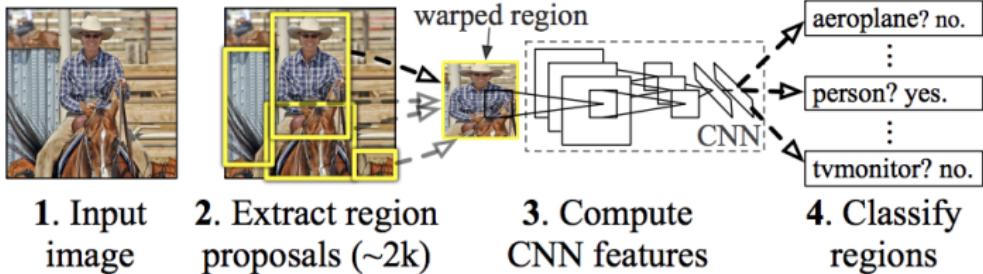
| Network            | Layers | Top-1 error | Top-5 error | #params | Year |
|--------------------|--------|-------------|-------------|---------|------|
| AlexNet            | 8      | 42.9%       | 19.8%       | 60M     | 2012 |
| VGG-16             | 16     | 27.0%       | 8.8%        | 138M    | 2014 |
| Inception V2       | 22     | 25.2%       | 7.8%        | 11M     | 2015 |
| ResNet-152         | 152    | 21.4%       | 5.7%        | 60M     | 2015 |
| ResNeXt-101 32x4   | 101    | 19.1%       | 4.4%        | 84M     | 2017 |
| Meta Pseudo Labels | 800+   | 9.8%        | 1.2%        | 480M    | 2021 |

- In recent years, much focus on refining existing models.
- The best models today require dedicated supercomputers.
  - Apply large scale architecture search for best hyperparameters.
  - Ensembles - train multiple classifiers and combine into a new one.
  - Semi-supervised - add billions of automatically annotated images.

# Object detection with CNNs

- Much harder problem – first need to find object candidates.

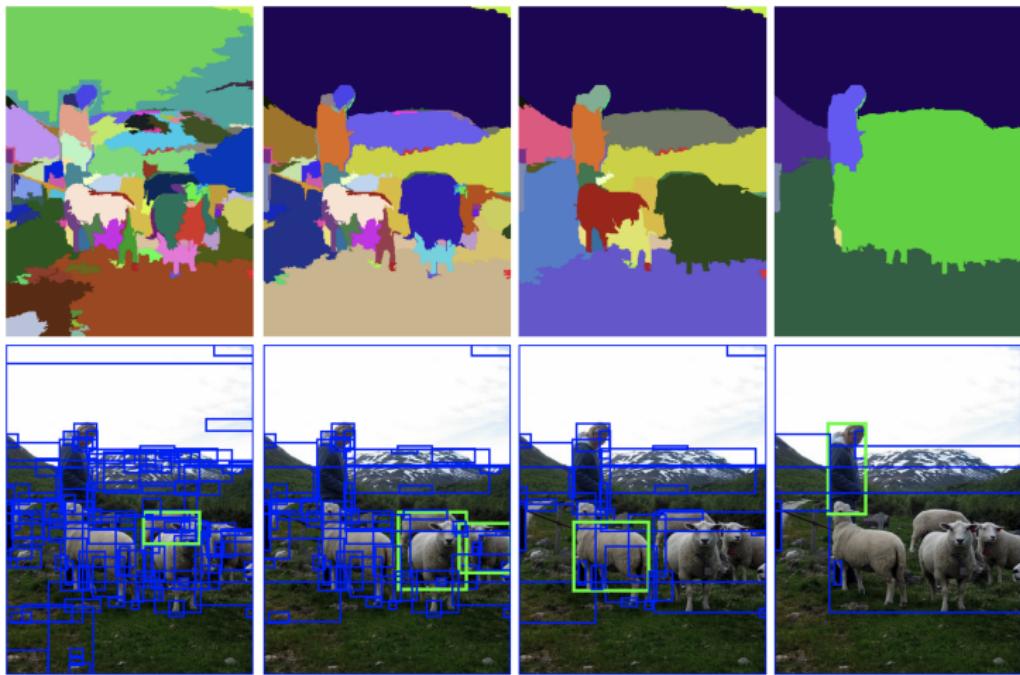
## R-CNN: *Regions with CNN features*



- Slow, since each region is warped and processed by network.

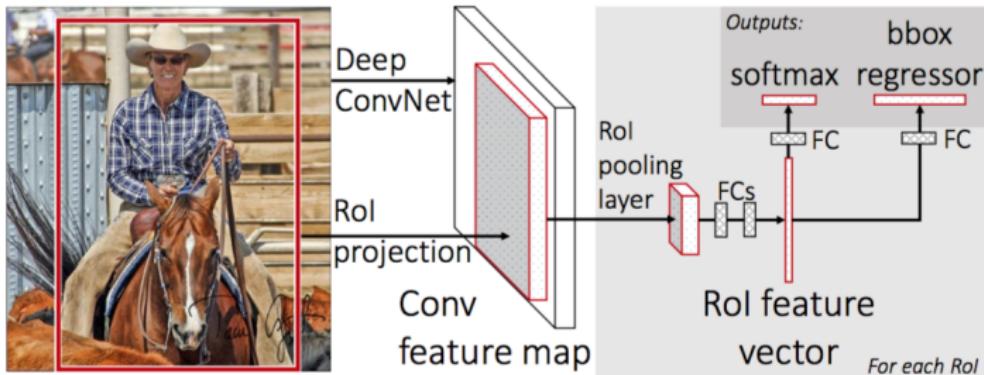
Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation.", CVPR 2014.

# Selective search for R-CNN (2014)



Apply Mean-Shift segmentation at different scales and then hierarchically merge regions to get object candidates.

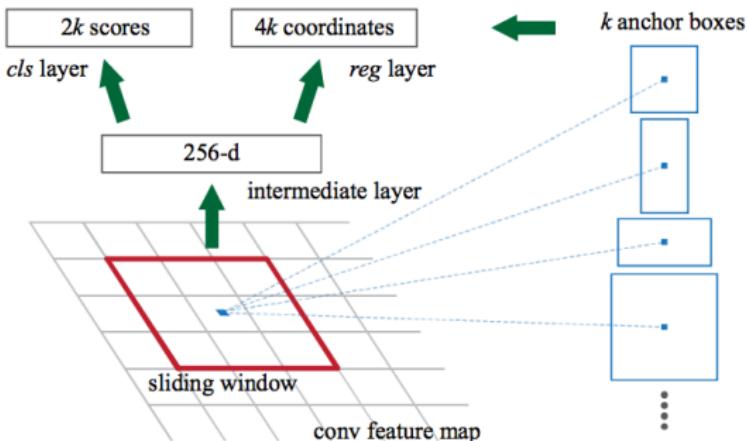
# Fast R-CNN (2015)



- Novelty: do convolutions only on the full image.
- With pooling, apply region proposals directly to feature space.
- Refine object by a bounding box regressor.

Girshick, "Fast R-CNN", ICCV 2015.

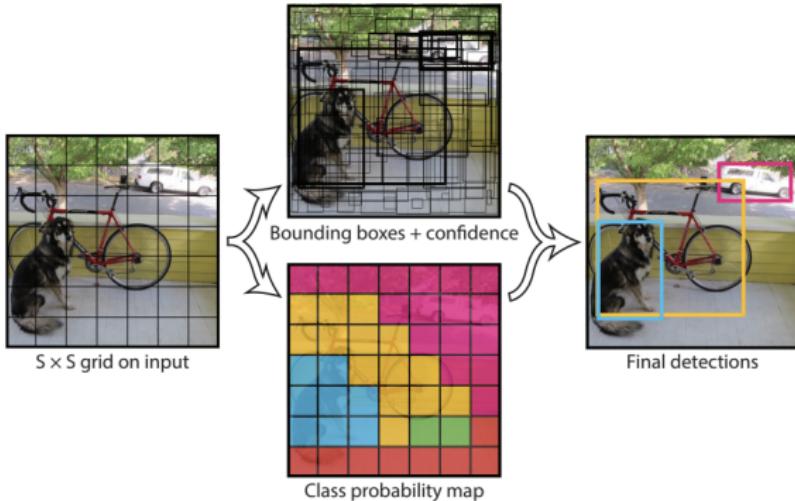
# Faster R-CNN (2015)



- Novelty: replace selective search (mean-shift) with a CNN.
- First compute feature map, then test different positions and sizes.
- Each anchor outputs class scores and refined bounding boxes.

Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks.", NIPS 2015.

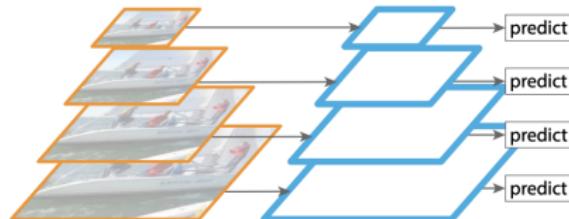
# YOLO – You only look once (2016)



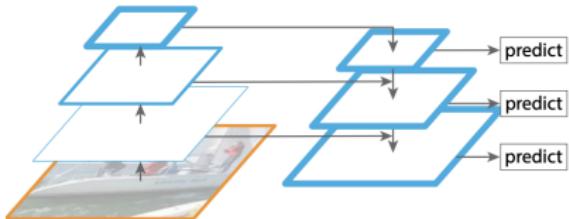
- Novelty: decouple class score from bounding box prediction.
- Combine generation and classification in one pass → much faster
- Uses fewer anchor points, leads to problems with small objects.

Redmon et al. "YOLO: Unified, real-time object detection", CVPR 2016.

# FPN – Feature pyramid networks (2016)



(a) Featurized image pyramid

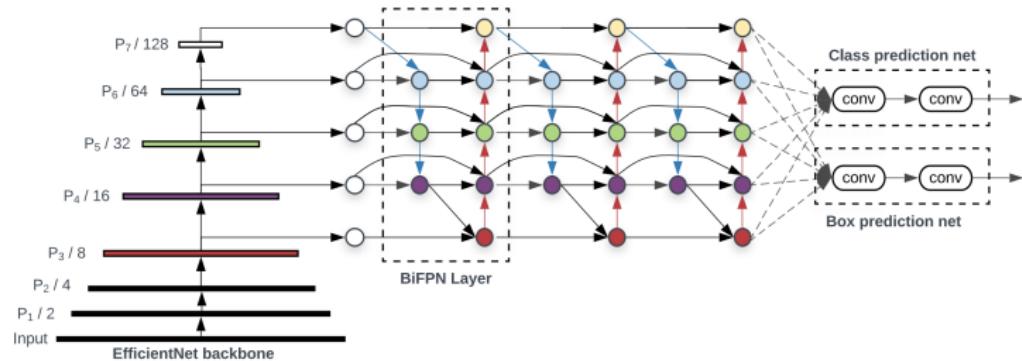


(d) Feature Pyramid Network

- Novelty: Faster R-CNN run at different scales in a pyramid.
- Instead of a Gaussian pyramid (left), use a pyramid of features (right) and combine features of different scales.

Lin et al., "Feature pyramid networks for object detection", CVPR 2017.

# EfficientDet (2020)



- Novelty: Uses 1) EfficientNet as a base network and 2) a more complex method of combining features at different scales.
- EfficientNet: for a given time budget, the image size, number of channels and layers are optimized using architecture search.

Tan et al, "EfficientDet: Scalable and efficient object detection", CVPR 2020.

# Object detection precision and speed

## Results on PASCAL VOC07 benchmark

| Method       | Accuracy (mAP) | FPS  | Year |
|--------------|----------------|------|------|
| R-CNN        | 68.4           | 0.02 | 2014 |
| Fast R-CNN   | 70.0           | 0.44 | 2015 |
| Faster R-CNN | 76.4           | 5    | 2015 |
| YOLOv1       | 63.4           | 45   | 2016 |
| YOLOv2       | 78.6           | 34   | 2017 |

## Results on COCO benchmark

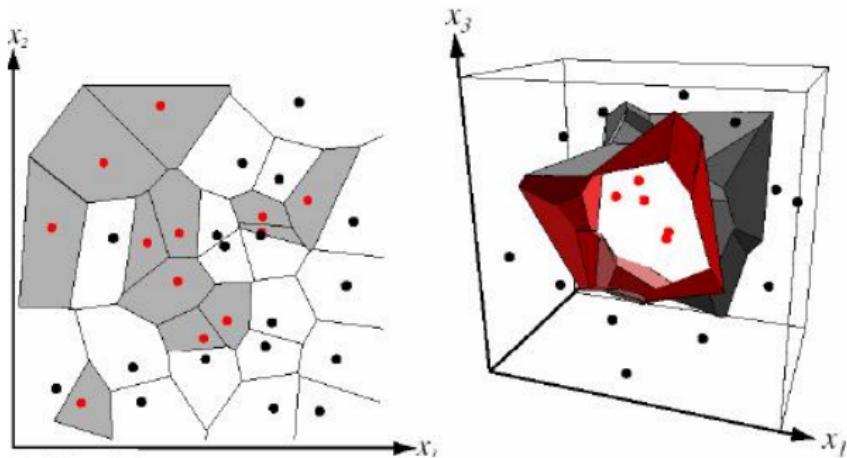
| Method             | Accuracy (box AP) | FPS | Year |
|--------------------|-------------------|-----|------|
| Faster R-CNN       | 34.7              | 5   | 2015 |
| Faster R-CNN + FPN | 36.2              | 5   | 2017 |
| EfficientDet-D7    | 53.7              | 6   | 2020 |

Since the original papers, significant improvements have been made to methods above. Optimization with respect to either speed or accuracy.

How to determine which class a feature vector belongs to?

- Nearest neighbour classification
- Bayesian classification
- Linear discriminant functions

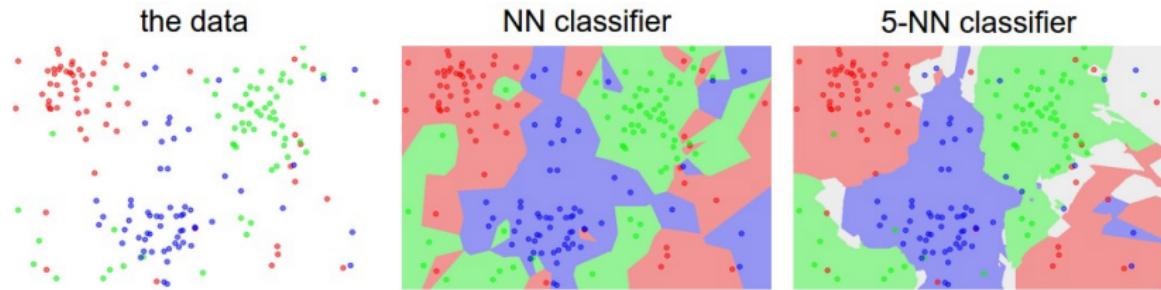
# Nearest neighbour classifier



- Straight-forward (model-less) method: given a test image, find the nearest neighbour among all training examples.
- Problem: Number of training examples can be very large and search for nearest neighbour too costly.

- Measure feature vectors for a representative selection of images from known classes.
- Given a feature vector, let the classification be the class index of the nearest representative in the scatter diagram.
  - + Works for well separated classes.
  - + Can represent clusters with complex shape.
  - May require complex computations in higher dimensions.
  - Depends on choice of metrics.
  - Highly sensitive to outliers (no suppression).

# K-Nearest neighbour classification



- Pick the class with most votes among K-nearest neighbours.
- More robust to individual weird samples (outliers).

- Idea: consider the class assignment and feature vectors as stochastic variables. Determine a classification function that minimizes the classification error.
- If we know the prior probability of the class,  $p(k)$ , and distribution of feature vector for class  $k$ ,  $p(z | k)$ , we know class probability given the feature vector,  $p(k | z)$ .

$$p(z, k) = p(z | k) p(k) = p(k | z) p(z) \Rightarrow$$

$$p(k | z) = \frac{p(z | k) p(k)}{p(z)} = \frac{p(z | k) p(k)}{\sum_i p(z | k_i) p(k_i)} \quad (\text{Bayes' formula})$$

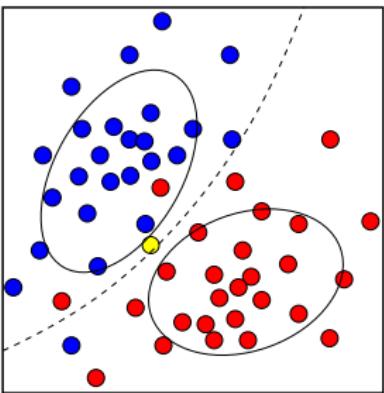
Choose the class  $k$  that maximizes  $p(k | z)$ .

- Minimizes the probability of wrong classification, **IF** statistical model is correct.
- Given most probable class, if prior probabilities  $p(k)$  are known.
- Common assumption: All  $p(k)$  equal  $\Rightarrow$  Maximum likelihood

# Assuming 2D Gaussian distributions

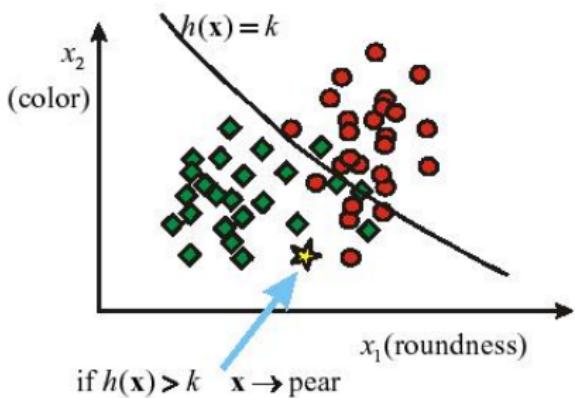
$$p(\mathbf{z} | k_i) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{z} - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1} (\mathbf{z} - \boldsymbol{\mu}_i) \right\}$$

$$p(k_1 | \mathbf{z}) = \frac{p(\mathbf{z} | k_1) p(k_1)}{p(\mathbf{z} | k_1) p(k_1) + p(\mathbf{z} | k_2) p(k_2)}$$



Decision boundary:  $p(\mathbf{z} | k_1) p(k_1) = p(\mathbf{z} | k_2) p(k_2)$

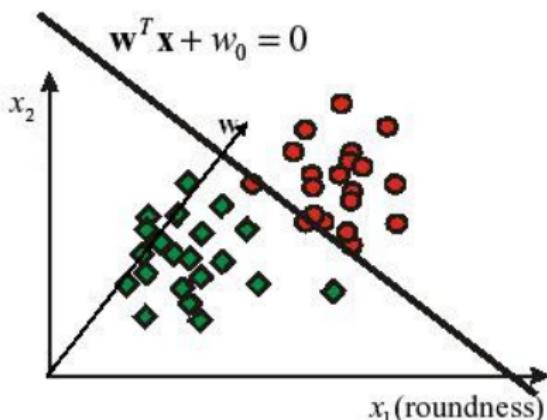
# Discriminant functions



Instead of modelling the distributions and then find a decision function, find a decision function and its boundary directly.

1. Choose class of decision function.
2. Estimate parameters of this function from training data.
3. Classify a new point based on the decision function.

# Linear Discriminant Functions



- Problem: Finding the best  $\mathbf{v} = (\mathbf{w}, w_0)$  given training examples.
- Most modern machine learning methods (Boosting, SVM, FCNN) use combinations of many linear discriminant functions.
- Note that the last fully-connected layers of a CNN can be seen as a combination of linear discriminant functions.

# Summary of good questions

- What is the difference between recognition and classification?
- What makes a good feature space for recognition?
- What kind of invariances do you often want in recognition?
- What classes of recognition methods exist and what are their differences?
- What does a typical feature based method consist of?
- What steps does a Bag of Words approach include?
- What are the dominating methods based on deep learning?
- What characteristics does a nearest neighbour classifier have?
- How do you find a decision boundary for Bayesian classification?

- Gonzalez & Woods: Chapter 12.4
- Szeliski: Chapter 5.1 and 6.2-6.3