Segmentation

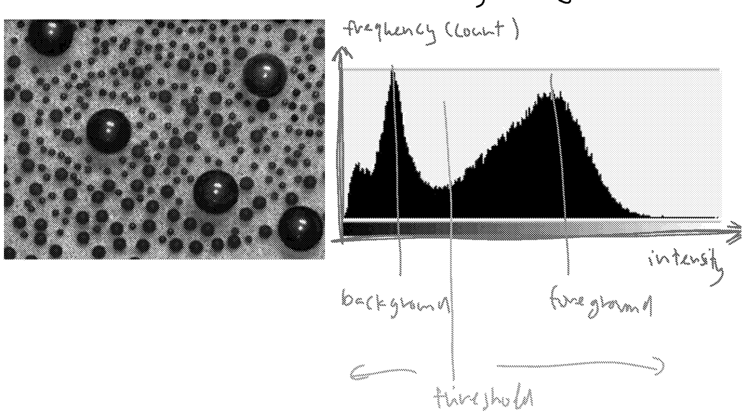
* split image into sub-parts:
  1) Based on color
  2) Based on spatial location
  3) Based on features
  4) Based on semantics

Problem statement

- Separate object(s) from background
- Find contours of objects
- Semantic image segmentation: label each pixel in the image with class label

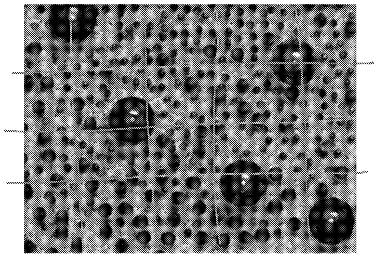
Color segmentation

Intensity histogram



Color segmentation

- Break image into blocks
- Find threshold in each block
- interpolate threshold values to create segmentation surface
- At each pixel location compare image intensity to segmentation surface.



Spatial context

Agglomerative segmentation:

* start with each pixel in a separate cluster
* Merge clusters with small distance
* Repeat while clusters are not satisfactory

Divisive segmentation

* Start with all pixels in one cluster
* split clusters to produce large distance between them
* Repeat while clusters are not satisfactory

Feature based segmentation

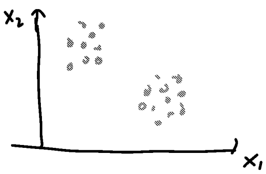
* Define feature vector at each pixel x:

$$F(x) = \begin{bmatrix} x \\ I(x) \\ L(x) \end{bmatrix} \begin{matrix} \leftarrow \text{location} \\ \leftarrow \text{intensity} \\ \leftarrow \text{local characteristics (e.g. texture)} \end{matrix}$$

* Apply clustering

Clustering

- Each pixel is assigned a feature vector (computed)
- To define a cluster group pixels with similar feature vectors.
- The similarity in each cluster should be high compared to the similarity between clusters.



Clustering

Fundamental problem:

* Need to compute cluster parameters and assign items to clusters
* Joint estimation is difficult.

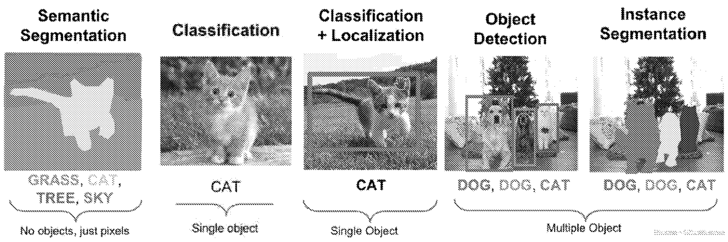
Issues:

* what is the number of clusters?
* How to compute similarity within clusters and distance between clusters?
* what features to use?

Clustering

* Algorithms:
  - k-means
  - Mixture of Gaussians
  - Mean shift
  - Expectation maximization
  - Graph cuts
  - spectral clustering

Classification tasks



Object recognition

* semantic gap (pixel vs labels):
  - pixels values as descriptors are sensitive to small variations
* object recognition needs to be invariant
  - pose (rotation, translation)
  - illumination
  - deformation (e.g. articulated objects)
  - occlusion
  - background
  - Natural variability (e.g. cars)

Object recognition

* want:
  - generalization (invariance)
  - extend to other problems
* Need:
  - feature extraction (invariance)
  - classification algorithm

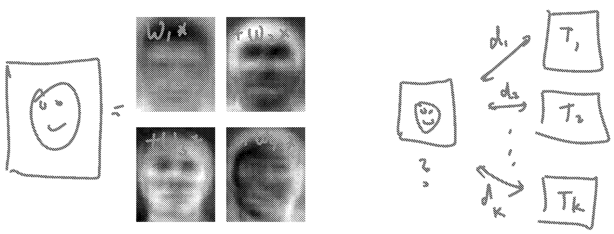
Object recognition

1) extract features (e.g. SIFT, HoG) and train a classifier.
2) Bag-of-words: extract patches and cluster them to form a "code book". Describe images using similarity to code words (with or without count).
3) Eigen faces: map images to a lower dimensional space where less sensitivity to variations and classify in this domain
4) Convolutional Neural Networks (CNN): learn feature extraction and classification.

Object recognition

* Eigenfaces approach:

- map image (e.g. 100x100) to lower dimensional vectors (e.g. 64) using PCA
- measure similarity to templates in lower dimensional space (less sensitive to small variations)



Object recognition

* Bag-of-words:
  - extract features (e.g. SIFT or HoG)
  - cluster features to create a codebook (dictionary)
  - Compute a distribution of code words in each class
  - classify using distribution of (code words)

Convolutional neural networks

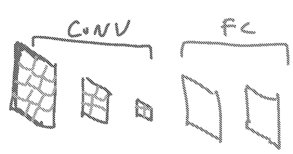
* Convolutional Neural Networks (CNN): learn feature extraction and classification.

* Learn from examples instead of coding algorithms

* Architecture for one class can work for others by changing data → class generalization

Convolutional neural networks

- Learn filters instead of specifying them using data (e.g. instead of edge detection)
- Combine features at multiple scales (multiple scale analysis)
- Produce decisions based on lower level features.

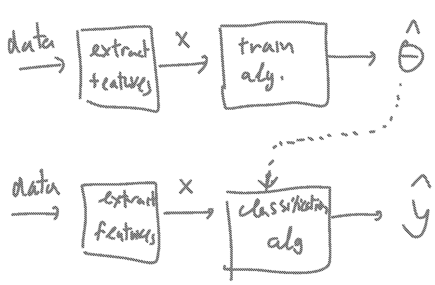


Data-driven recognition

* Training and inference:

Training:

Inference:



Data-driven recognition

- Train/test on different collection (e.g. K-fold cross validation)
- Performance based on testing error (generalization vs. over-fitting)
- Large models have more capacity but require more data (exponentially) and may over-fit.
- parameters:
  - Learn model parameters (training).
  - Hyper-parameters are specified parameters