EEE422/6082 Computational Vision

Image Categorization

Ling Shao

Many slides from Derek Hoiem

Last classes

- Object recognition: localizing an object instance in an image
- Face recognition: matching one face image to another

Today's class: categorization

- Overview of image categorization
- Representation
  – Image histograms
- Classification
  – Important concepts in machine learning
  – What the classifiers are and when to use them

Image Categorization

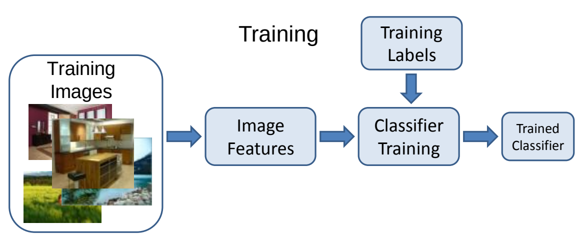
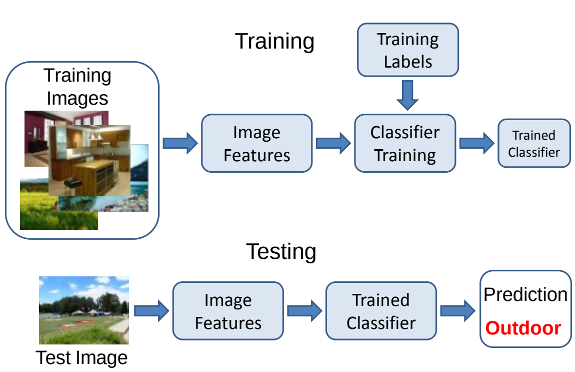
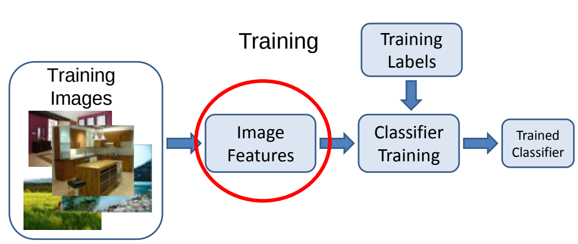


Image Categorization



Part 1: Image features

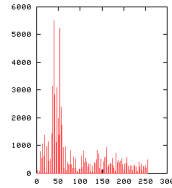


General Principles of Representation

- Coverage
  - Ensure that all relevant info is captured
- Concision
  - Minimize number of features without sacrificing coverage
- Directness
  - Ideal features are independently useful for prediction

Image Intensity

Image Representations: Histograms



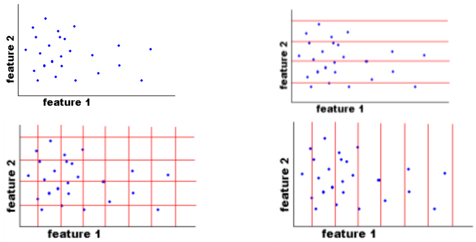
Global histogram

- Represent distribution of features
  - Color, texture, depth, …

Images from Dave Kauchak

Image Representations: Histograms

Histogram: Probability or count of data in each bin



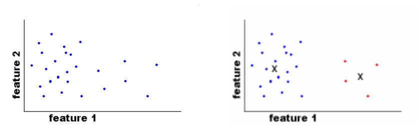
- Joint histogram
  - Requires lots of data
  - Loss of resolution to avoid empty bins

Marginal histogram
- Requires independent features
- More data/bin than joint histogram

Images from Dave Kauchak

Image Representations: Histograms

Clustering



Use the same cluster centers for all images

Images from Dave Kauchak

Computing histogram distance

$$\mathrm{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2}\sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared

Histograms: Implementation issues

- Quantization
  - Grids: fast but only applicable with few dimensions
  - Clustering: slower but can quantize data in higher dimensions

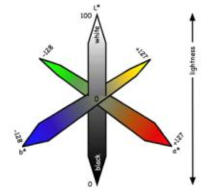
Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation

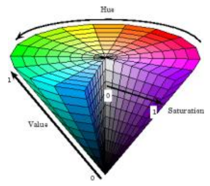
- Matching
  - Histogram intersection or Euclidean may be faster
  - Chi-squared often works better
  - Earth mover's distance is good for when nearby bins represent similar values

What kind of things do we compute histograms of?

- Color



L*a*b* color space

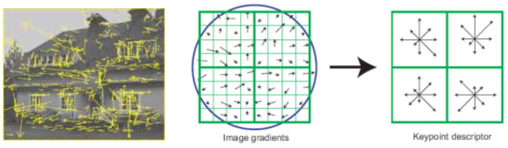


HSV color space

- Texture (filter banks or HOG over regions)

What kind of things do we compute histograms of?

- Histograms of gradient



SIFT – Lowe IJCV 2004

- Visual words

Image Categorization: Bag of Words

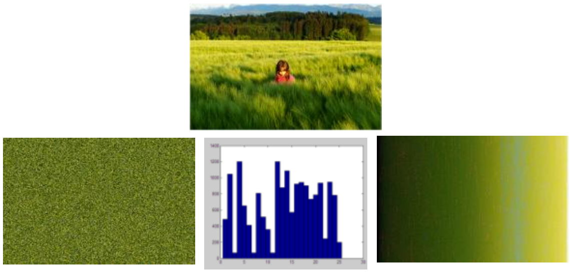
Training

1. Extract keypoints and descriptors for all training images
2. Cluster descriptors
3. Quantize descriptors using cluster centers to get "visual words"
4. Represent each image by normalized counts of "visual words"
5. Train classifier on labeled examples using histogram values as features

Testing

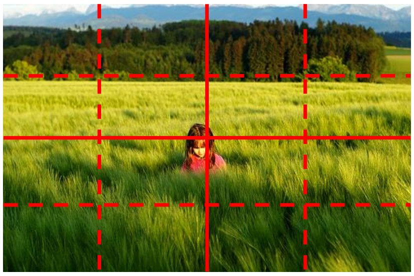
1. Extract keypoints/descriptors and quantize into visual words
2. Compute visual word histogram
3. Compute label or confidence using classifier

But what about layout?



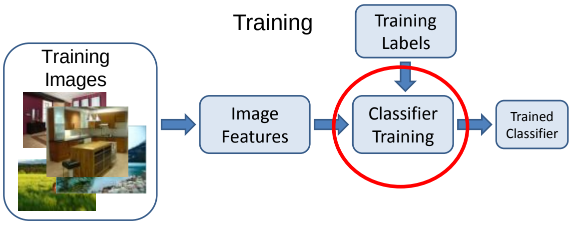
All of these images have the same color histogram

Spatial pyramid



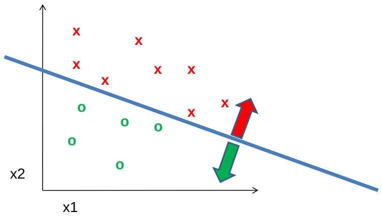
Compute histogram in each spatial bin

Part 2: Classifiers



Learning a classifier

- Given some set features with corresponding labels, learn a function to predict the labels from the features



Many classifiers to choose from

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- Etc.

Which is the best one?

No Free Lunch Theorem

The perfect classification algorithm

- Objective function: solves what you want to solve
- Parameterization: makes assumptions that fit the problem
- Regularization: right level of regularization for amount of training data
- Training algorithm: can find parameters that maximize objective on training set
- Inference algorithm: can solve for objective function in evaluation

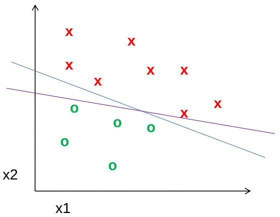
Generative vs. Discriminative Classifiers

Generative
- Training
  - Maximize joint likelihood of data and labels
  - Assume (or learn) probability distribution and dependency structure
  - Can impose priors
- Testing
  - P(y=1, x) / P(y=0, x) > t?
- Examples
  - Foreground/background GMM
  - Naïve Bayes classifier
  - Bayesian network

Discriminative
- Training
  - Learn to directly predict the labels from the data
  - Assume form of boundary
  - Margin maximization or parameter regularization
- Testing
  - $f(x) > t$; e.g., $w^T x > t$
- Examples
  - Logistic regression
  - SVM
  - Boosted decision trees

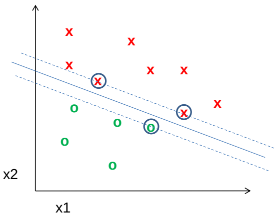
Classifiers: Linear SVM

- Objective
- Parameterization
- Regularization
- Training
- Inference



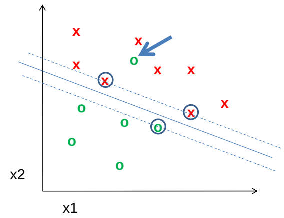
Classifiers: Linear SVM

- Objective
- Parameterization
- Regularization
- Training
- Inference



Classifiers: Linear SVM

- Objective
- Parameterization
- Regularization
- Training
- Inference

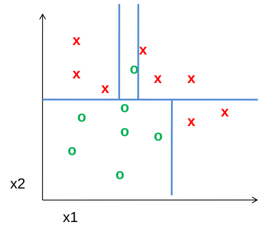


Using SVMs

- Good general purpose classifier
  - Generalization depends on margin, so works well with many weak features
  - No feature selection
  - Usually requires some parameter tuning
- Choosing kernel
  - Linear: fast training/testing – start here
  - RBF: related to neural networks, nearest neighbor
  - Chi-squared, histogram intersection: good for histograms (but slower, esp. chi-squared)
  - Can learn a kernel function

Classifiers: Decision Trees

- Objective
- Parameterization
- Regularization
- Training
- Inference



Ensemble Methods: Boosting

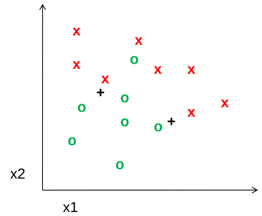
Discrete AdaBoost(Freund & Schapire 1996*b*)

1. Start with weights $w_i = 1/N$, $i = 1, \dots, N$.
2. Repeat for $m = 1, 2, \dots, M$:
   (a) Fit the classifier $f_m(x) \in \{-1, 1\}$ using weights w_i on the training data.
   (b) Compute $\mathrm{err}_m = E_w[1_{(y \neq f_m(x))}]$, $c_m = \log((1 - \mathrm{err}_m)/\mathrm{err}_m)$.
   (c) Set $w_i \leftarrow w_i \exp[c_m \cdot 1_{(y_i \neq f_m(x_i))}]$, $i = 1, 2, \dots N$, and renormalize so that $\sum_i w_i = 1$.
3. Output the classifier $\mathrm{sign}[\sum_{m=1}^{M} c_m f_m(x)]$

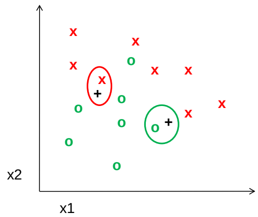
figure from Friedman et al. 2000

K-nearest neighbor

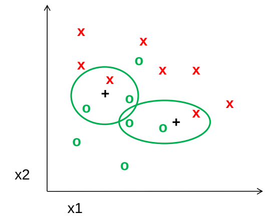
- Objective
- Parameterization
- Regularization
- Training
- Inference



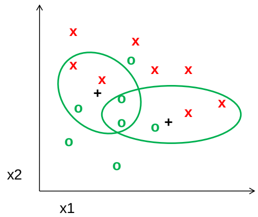
1-nearest neighbor



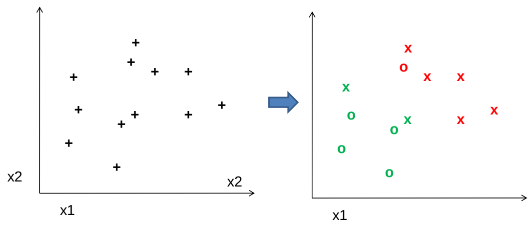
3-nearest neighbor



5-nearest neighbor



Clustering (unsupervised)



What to remember about classifiers

- No free lunch: machine learning algorithms are tools, not dogmas
- Try simple classifiers first
- Better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data (bias-variance tradeoff)

Some Machine Learning References

- General
  - Tom Mitchell, *Machine Learning*, McGraw Hill, 1997
  - Christopher Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995
- Adaboost
  - Friedman, Hastie, and Tibshirani, “Additive logistic regression: a statistical view of boosting”, Annals of Statistics, 2000
- SVMs
  - http://www.support-vector.net/icml-tutorial.pdf