Introduction to Scene Classification

Jia Xu
jiaxu@cs.wisc.edu
Outline

➢ Introduction
➢ Key Components for Scene Classification
➢ Locality-constrained Linear Coding (LLC)
➢ Experimental Evaluation
➢ Discussion
Human Vision

What is the scene?
- office? street?
suburb?

How do you recognize this scene?

http://alfi.soils.wisc.edu/asig/webcam/big.jpg
Computer Recognition/Classification

- **Hand-designed Feature Extraction**
- **Trainable Classifier**

- **Image** ➔ **Scene Class**

- **Low level features**
  - Color histogram, SIFT, HOG, Object Bank, e.t.c

- **Feature engineering**
  - Bag of Words, Spatial Pyramid Matching (SPM), Sparse Coding(SC), Locality-constrained Linear Coding(LLC)

- **Classifier**
  - SVM (linear, nonlinear, multiple kernel), linear/logistic regression, random forest
Image Classification Overview

[Image: Diagram showing the process of image classification with stages such as Local Gradients, Pooling, VQ Coding, Average Pooling, and Output Labels, with examples like SIFT, HOG.]

[Source: K. Yu, R. Urtasun]
Feature Extraction

For each image, we get $N$ feature points: $x_i, i=1, \ldots, N$.

Depending on sample scheme, $N$ can be different for different images.

[Source: S. Lazebnik]
Codebook Generation

➢ K-means
  ○ partition N features \( \{x_1, \ldots, x_N\} \) into K clusters \( \{b_1, \ldots, b_K\} \), where \( b_k \) is the center of k-th cluster
  ○ hard assignment

➢ Gaussian Mixture Model (GMM)
  ○ learn K Gaussian mixtures from the feature set \( \{x_1, \ldots, x_N\} \)
  ○ soft assignment

[Source: X. Wang et al.]
Feature Encoding

➢ Vector Quantization (VQ)

\[ c_{nk} = \begin{cases} 
1, & \text{if } k = \arg \min_k \| x_n - b_k \|^2 \\
0, & \text{otherwise} 
\end{cases} \]

➢ Soft-assignment Encoding

\[ c_{nk} = \frac{\exp(-\beta \| x_n - b_k \|^2)}{\sum_{j=1}^{K} \exp(-\beta \| x_n - b_j \|^2)} \]

Spatial context completely ignored!
Spatial Pyramid Representation

[Source: S. Lazebnik]
Pyramid Matching

Original images

Feature histograms:
Level 3

Level 2

Level 1

Level 0

Total weight (value of pyramid match kernel): \[ I_3 + \frac{1}{2}(I_2 - I_3) + \frac{1}{4}(I_1 - I_2) + \frac{1}{8}(I_0 - I_1) \]

(Source: S. Lazebnik)
Limitations of SPM

- Non-linear SVM is not scalable
- VQ coding may be too coarse
- Average pooling is not optimal

Why not non-linear coding and linear SVM?
LLC

Feature vector [ ]

Concating

SPM

Pooling

Code

Coding

Descriptor

Feature Extraction

Image

LLC Coding process

Step 3:
$c_i$ is an Mx1 vector with K non-zero elements whose values are the corresponding $c^*$ of step 2

input: $x_i$  →  code: $c_i$

Step 2:
Reconstruct $x_i$ using $B_i$

$$c^* = \arg\min_{c} \| x_i - c_i^T B_i \|^2$$

st. $\sum c_i = 1$

input: $x_i$

codebook: $B = \{b_i\}_{i=1,...,M}$

Step 1:
Find K-Nearest Neighbors of $x_i$, denoted as $B_i$

[Source: J. Wang et al.]
SC vs LLC

➢ Sparse Coding (SC)

\[ \arg \min_C \sum_{i=1}^{N} \| x_i - Bc_i \|^2 + \lambda \| c_i \|_{\ell^1} \]

➢ Locality-constrained Linear Coding (LLC)

\[ \min_{\tilde{c}} \sum_{i=1}^{N} \| x_i - \tilde{c}_i B_i \|^2 \]

\[ \text{st. } 1^T \tilde{c}_i = 1, \forall i. \]
clear; clc; close all;
N = 100; % feature dimension
K = 5; % number of nearest neighbours
% construct codebook
B = randn(K, N);
% create truth code
c = randn(K, 1);
c = c /sum(c);
% compute feature
x = B'*c;

% compute data covariance matrix
one = ones(K, 1);
B_1x = B - one *x';
C = B_1x * B_1x';

% reconstruct LLC code
c_hat = C \ one;
c_hat = c_hat /sum(c_hat);
% compute reconstruction error
diff = norm(c-c_hat)
Properties of LLC

- Better reconstruction
- Local smooth sparsity
- Analytical solution

[Source: J. Wang et al.]
Pooling and Normalization

Pooling
- sum pooling: $c_{out} = c_{in1} + \ldots + c_{in2}$
- max pooling: $c_{out} = \max(c_{in1}, \ldots, c_{in2})$

Normalization
- sum normalization: $c_{out} = c_{in} / \text{sum}(c_{in})$
- L2 normalization: $c_{out} = c_{in} / \|c_{in}\|_2$
Classification

➢ Linear SVM

○ Given training data $D=\{ (x_i,y_i) | x_i$ is a $p$ dimensional feature vector and $y_i = -1$ or $1$, $i=1,...,n$}\)

○ Solve for

$$\text{argmin}_{w,b} \frac{1}{2} ||w||_2$$

subject to (for any $i=1,...,n$)

$$y_i (w^{*}x_i - b) \geq 1$$

○ Various setting are implemented in liblinear
Multi-Class SVM

➢ Crammer and Singer algorithm
  ○ already implemented in liblinear

➢ One-vs-all
  ○ train C binary classifier \((w_c, b_c)\) for each category \(c\)
  ○ final label for image \(i\) is \(\text{argmax}_c \ w_c x_i + b_c\)
Dataset

➢ Scene category dataset from Lazebnik et al.
➢ 15 categories
➢ Each category has 200 to 400 images
➢ Average image size is 300*250
Experimental Evaluation Protocol

➢ Training with 100 images per class
  ○ codebook generation and linear SVM
➢ Testing with the rest images
➢ Report confusion matrix
➢ Report mean accuracy (mean of diagonal elements in your confusion matrix)
Evaluation Tips

➢ Only change one parameter each time, and save your results in a table
➢ Important parameters: # of codebook size (start with 1024), # of nearest neighbors (start with 5), -s -C -w in liblinear
➢ Grid search
Extensions

➢ Codebook Optimization
  ○ Algorithm 4.1
➢ Evaluate your implementation on other applications, e.g. action recognition.
➢ Combine LLC with Object Bank
Discussion

➢ Q & A