Automatic Image Orientation Detection

Mike Ghesquiere

CS9840 – Machine Learning for Computer Vision

Problem

Given an input image rotated by 0°, 90°, 180° or 270° detect the correct orientation

e.g. scanning vacation photos

Pipeline

Feature Extraction

Feature Selection

Learning

Feature Extraction

Split the image into $N \times N$ blocks ($N = 10$).

Extract features from each block.

- Color moments in L,U,V space
- Color histogram in H,S,V space
- Edge direction histograms
- MSAR texture features

Feature Extraction

For each image:

$\mathbf{10 \times 10}$ blocks ($\mathbf{100}$)

mean and variance ($\mathbf{2}$)

for each $\boldsymbol{L, U, V}$ channel ($\mathbf{3}$)

Gives a feature vector with $\mathbf{600}$ dimensions.

Pipeline

L, U, V Color moments

Feature Selection

Learning

Learning

Learning Vector Quantization (LVQ)

k-NN

SVM

Mixture of Gaussians

Hierarchical Discriminating Regression (HDR) tree

Vector Quantization

Given a ***codebook*** size q.

Vector Quantization aims to find q centroids with approximately the same number of points surrounding each.

Classification can then be done in a nearest neighbour fashion.

Learning Vector Quantization (LVQ) is a supervised counterpart (also nonparametric).

LVQ Training

Need a distance function:

$$dist(x, y) = \sqrt[2]{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \text{(Eucliean distance)}$$

…and a stop condition (e.g. max iterations, desired error rate).

For convenience, define:

$$i_{j,k} = \begin{cases} -1 & j \neq k \\ 1 & j = k \end{cases}$$

LVQ Training

Input: $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n$ training vectors with labels $y_1, y_2, \dots, y_n$

q number of codebook vectors ($q \geq supp(y_1, \dots, y_n)$)

α learning rate ($\alpha \in (0,1]$)

Output: $\overline{w}_1, \overline{w}_2, \dots, \overline{w}_q$ codebook vectors

Initialize $\overline{w}_1, \overline{w}_2, \dots, \overline{w}_q$ to a random subset of training vectors

while(not stop condition not met)

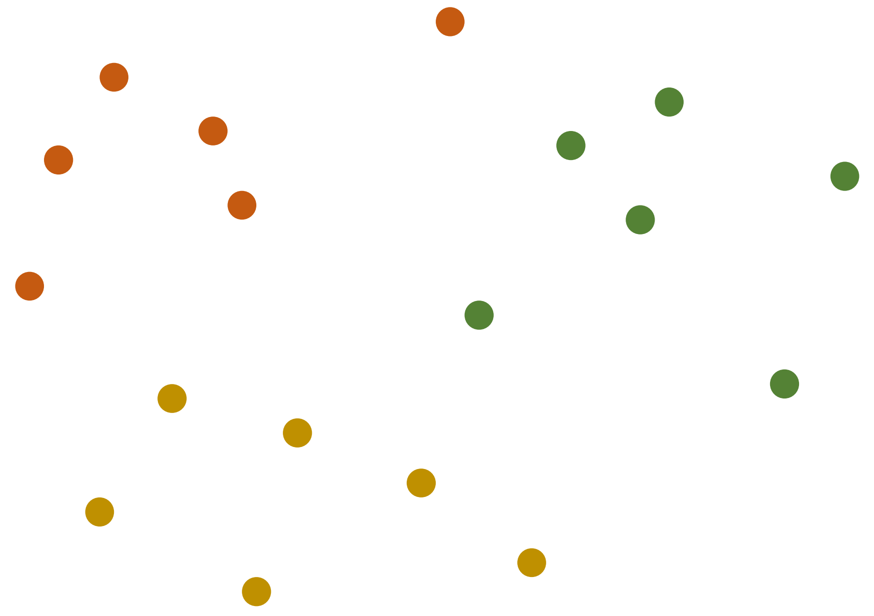
Select random $\bar{x}_i$

Select $\overline{w}_j$ such that $dist(\bar{x}_i, \overline{w}_j)$ is minimal

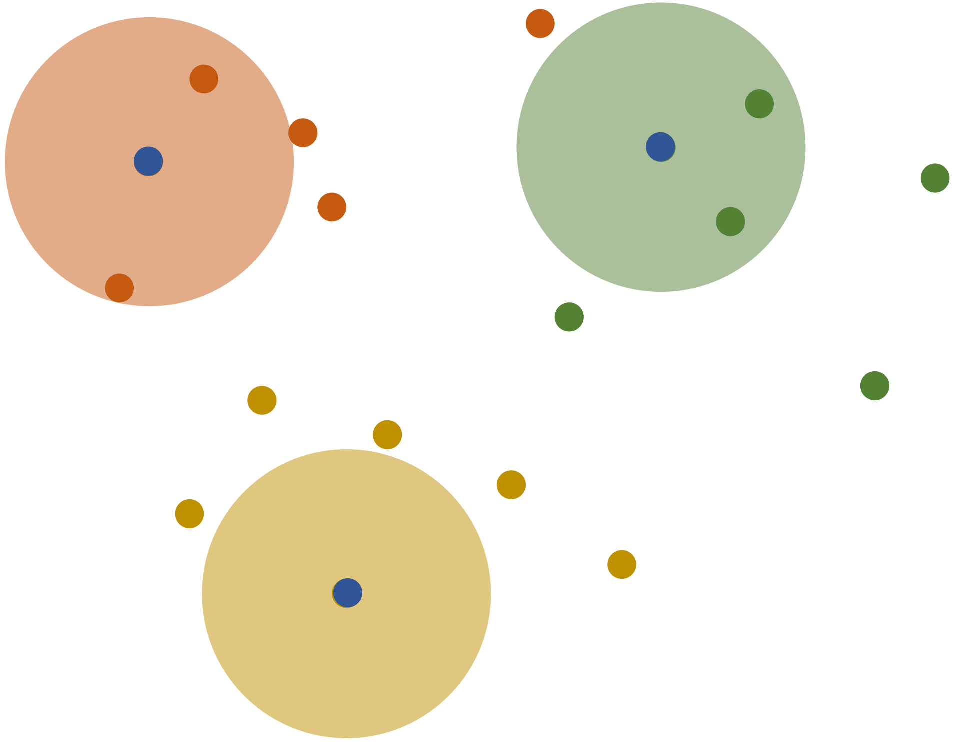
$\overline{w}_j \coloneqq \overline{w}_j + i_{y(\overline{w}_j), y(\bar{x}_i)} \alpha \left(\bar{x}_i - \overline{w}_j\right)$

return $\overline{w}_1, \overline{w}_2, \dots, \overline{w}_q$

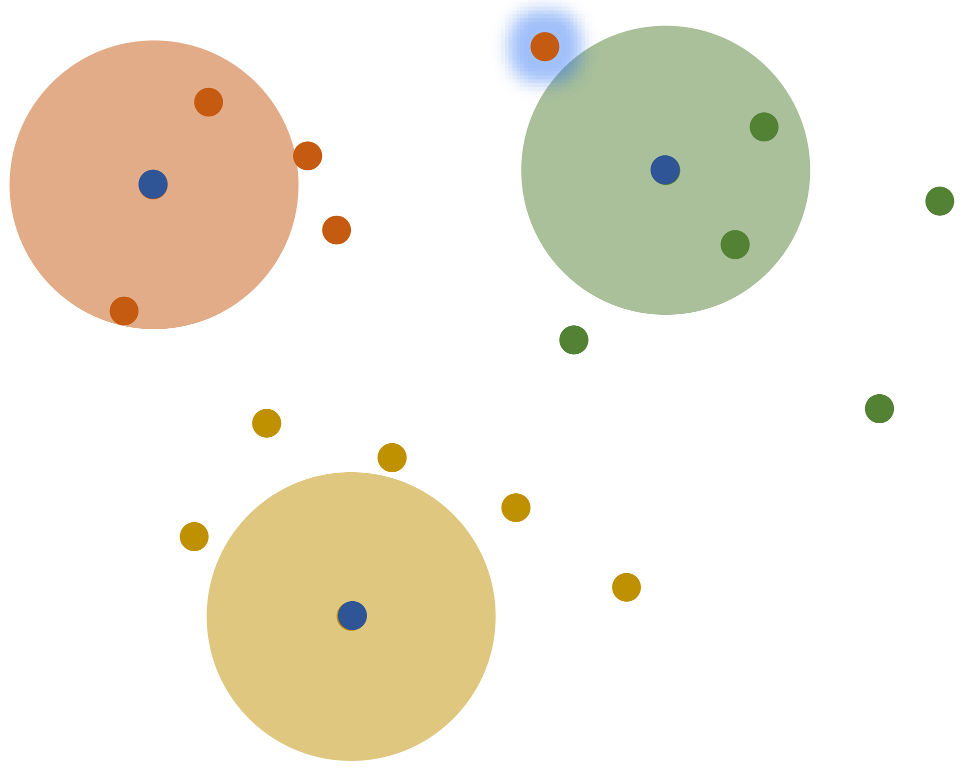
LVQ Training



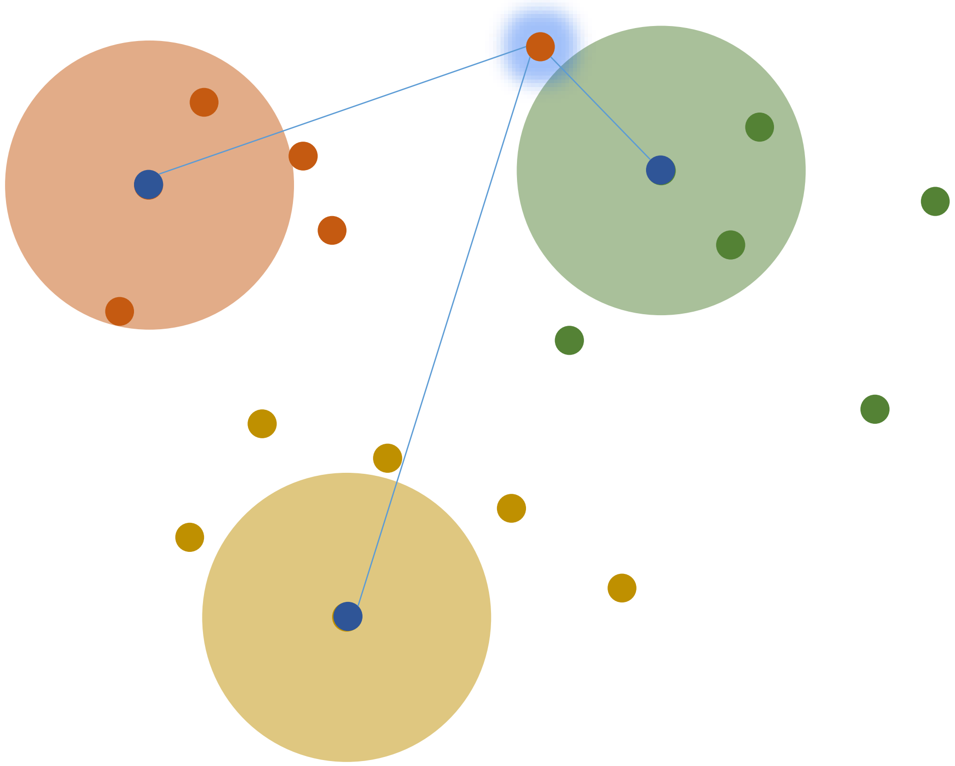
Initialize codebook vectors



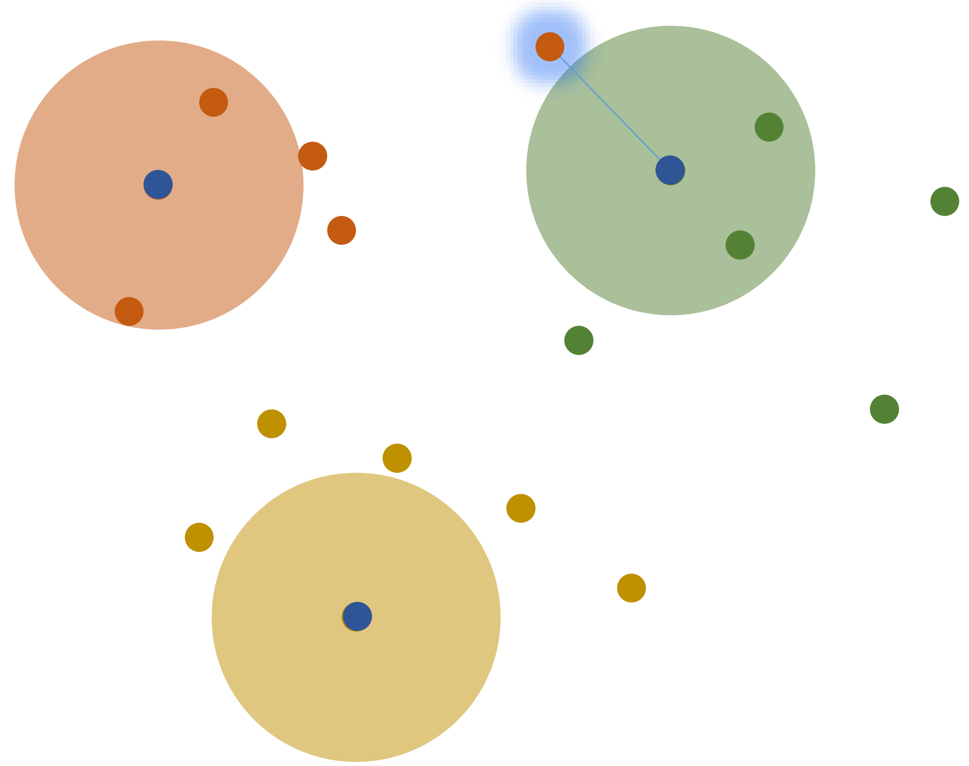
Select a training vector



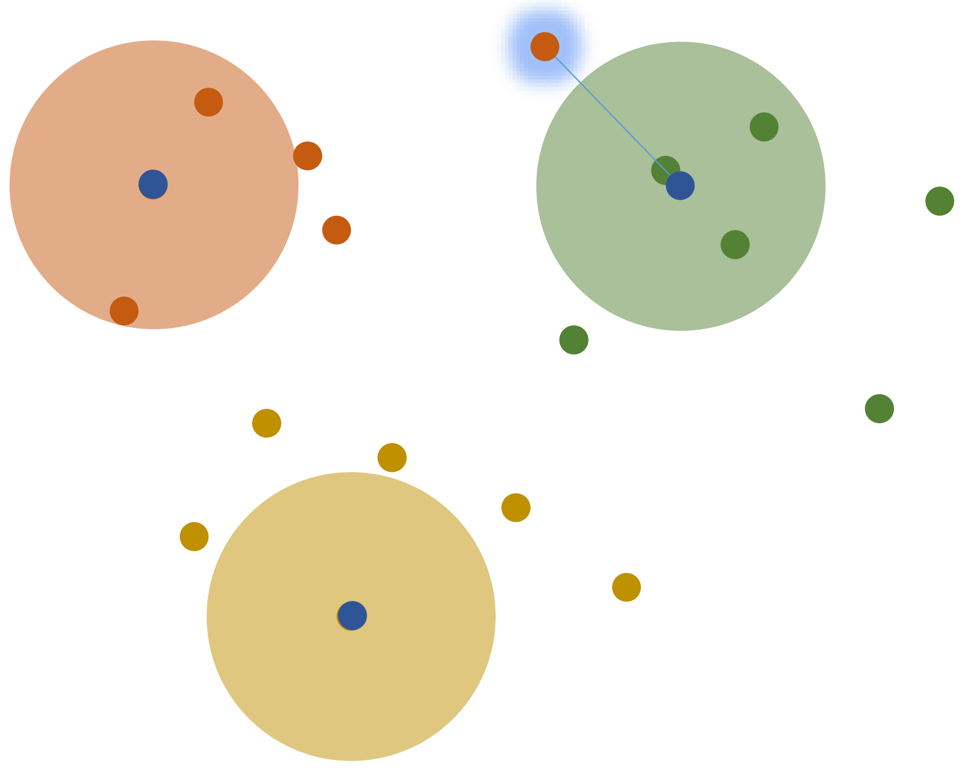
Find nearest codebook vector



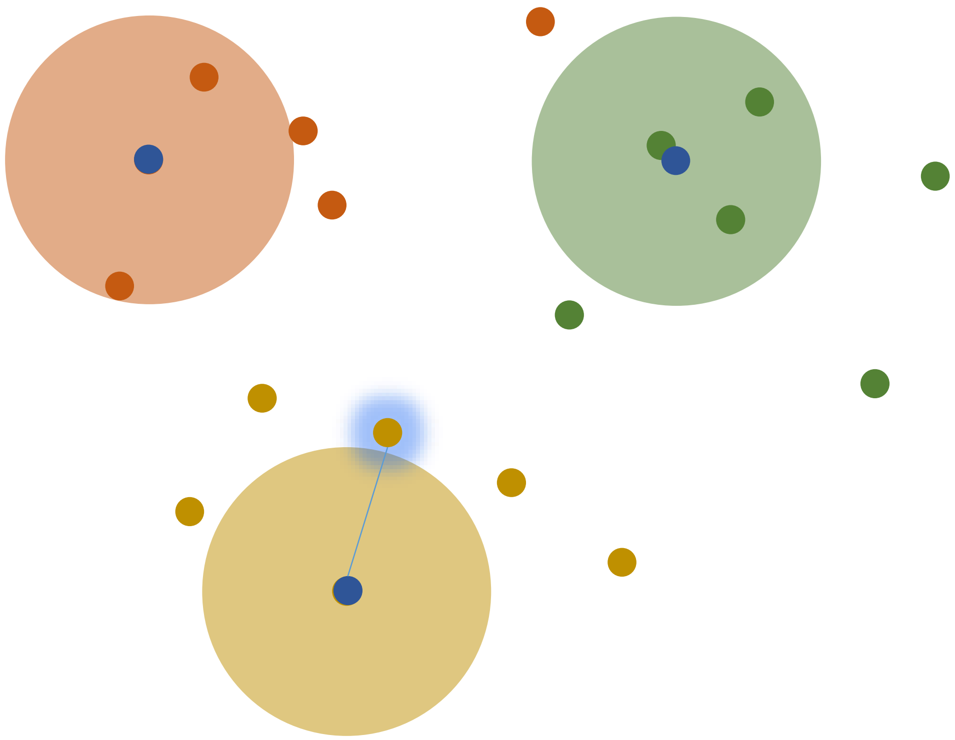
$$i_{y(\bar{w}_k),y(\bar{x}_i)} = -1$$



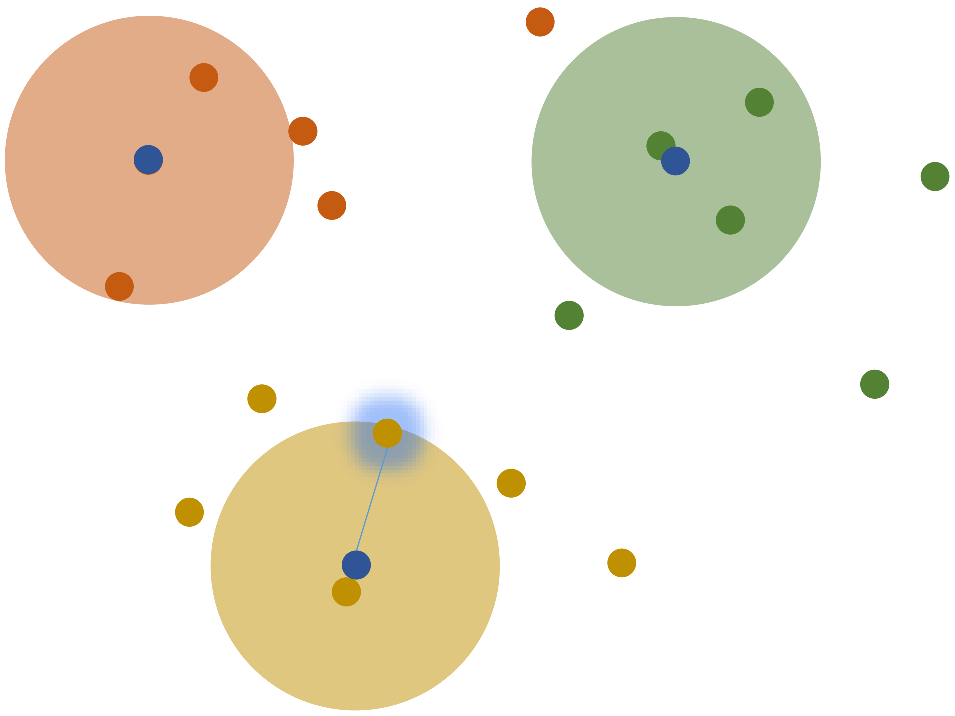
“push”



$$i_{y(\bar{w}_k),y(\bar{x}_i)} = 1$$



“pull”



Learning (cont'd)

q (codebook size) is not a learnable parameter.

Minimum Description Length (MDL):

$$\hat{q} = \arg\min_{q} \{ L(\mathcal{Y} | \theta_{(q)}) + L(\theta_{(q)}) \}$$

Modified MDL (MMDL):

$$\hat{q} = \arg\min_{q} \left\{ L(\mathcal{Y} | \theta_{(q)}) + \frac{q}{2} \log n + \frac{\dim(y^{(i)})}{2} \sum_{j=1}^{q} \log\left(m_j^{(i)} n\right) \right\}$$

Learning (cont'd)

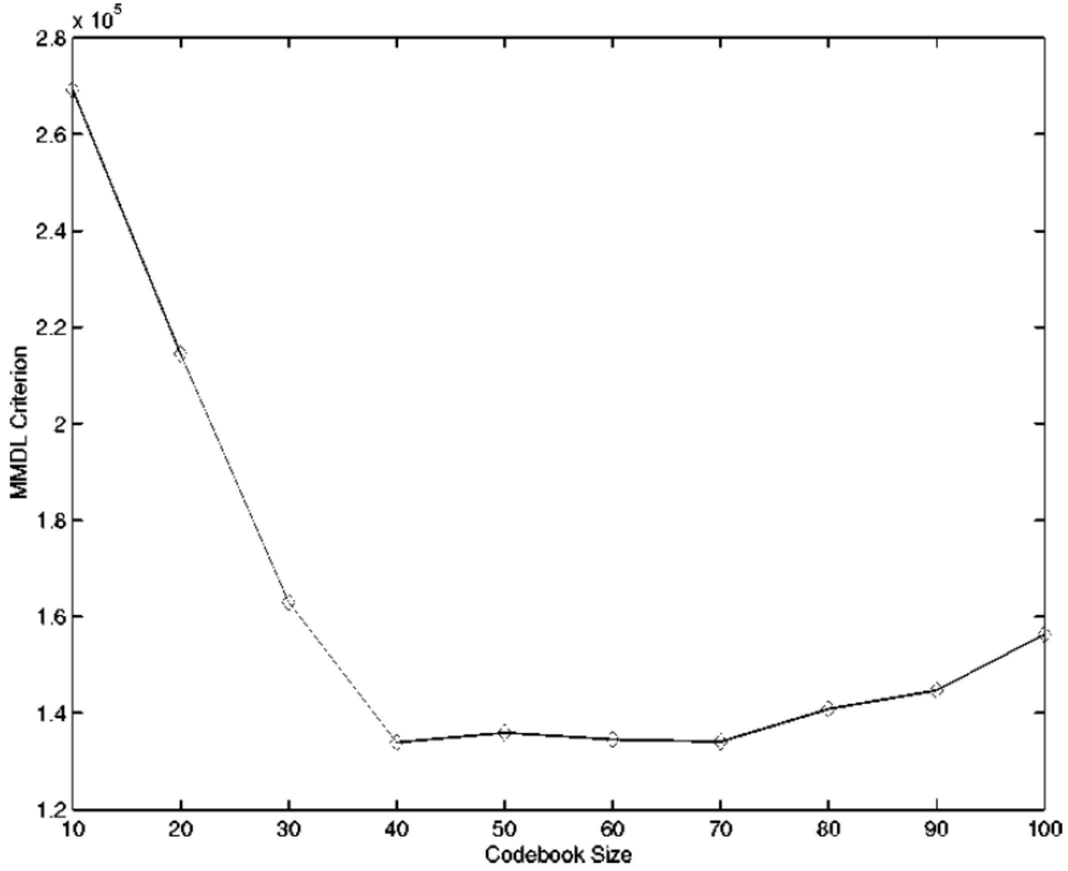


Fig. 3. Determining the optimal codebook size.

Pipeline

L, U, V Color moments

Feature Selection

LVQ

Feature Selection

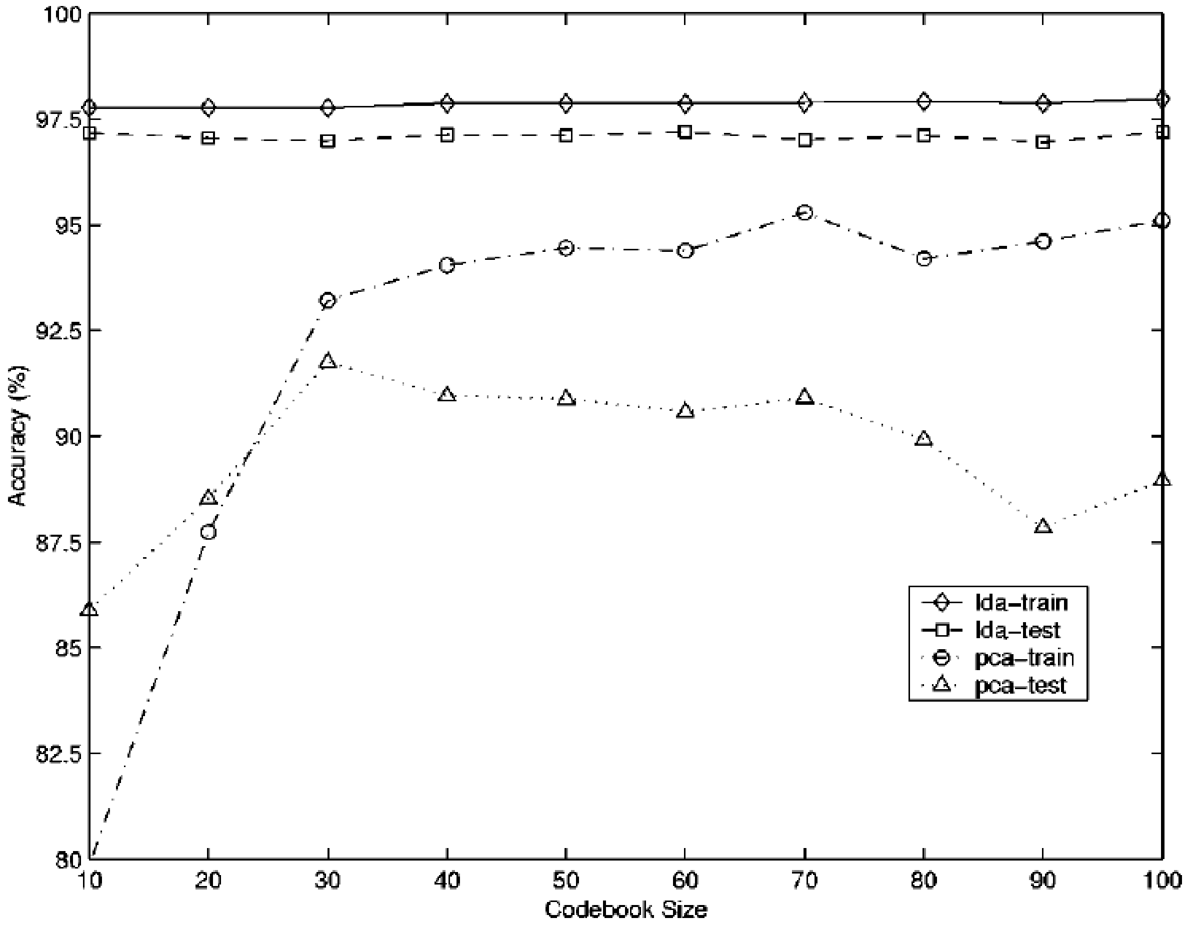


Fig. 6. Accuracy of LVQ-based classifier on features extracted using PCA and LDA.

Feature Selection

PCA

LDA

LDA (3-D projection)

Chernoff faces for LDA

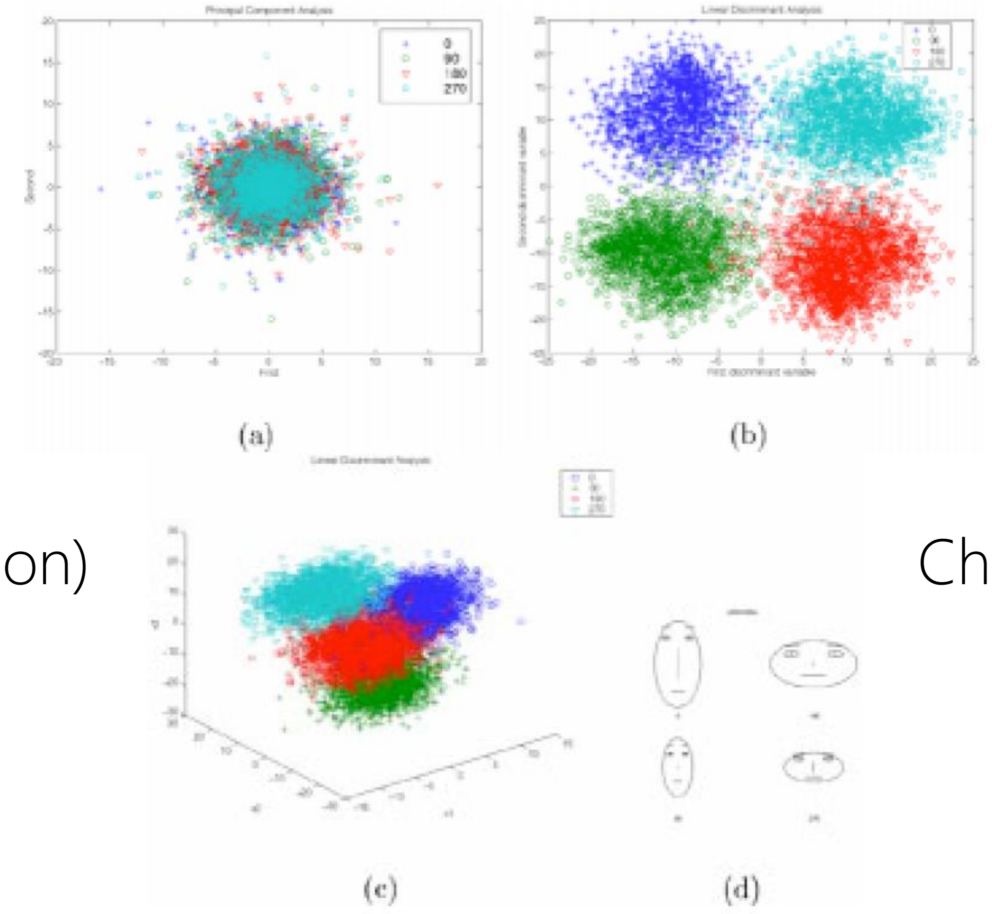


Fig. 4. Two-dimensional representation of the data using (a) PCA and (b) LDA. The original feature space is projected into 2-D space, along the largest two eigenvectors, and the first two discriminant variables, respectively. (c) Three-dimensional representation of the LDA space. (d) Chernoff faces corresponding to the mean vectors of the four categories in the 3-D LDA space.

K-NN variant selection

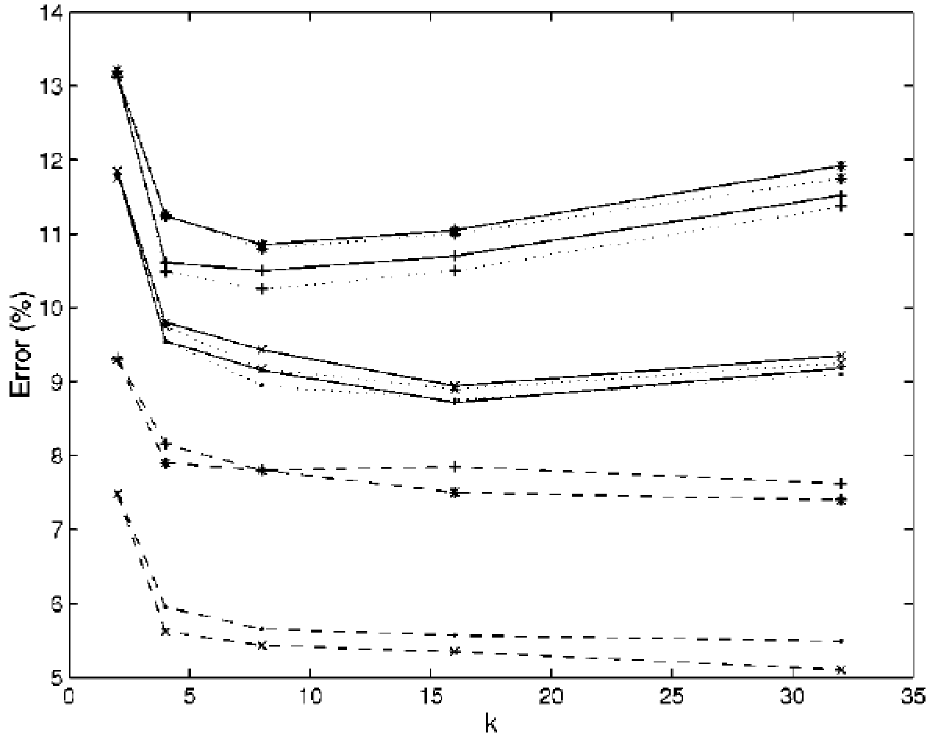


Fig. 5. Error rates of k-NN classifier w.r.t. k; scheme for classification are represented as follows: voting ($*$), distance weighted ($+$), voting with normalized data (x), and distance weighted with normalized data ($.$); results are shown before feature extraction (solid lines), feature extraction using PCA (dotted lines), and feature extraction using LDA (dashed lines).

Results

TABLE I
PERFORMANCE COMPARISON OF FIVE CLASSIFIERS

Method	Accuracy		Time (sec)	
	Training	Testing	Training	Testing (per image)
LVQ	98.6%	96.8%	410.00	0.005
LVQ with LDA	97.8%	97.2%	70.00	0.000025
Mixture of Gaussian	79.3%	88.4%	3,600.00	0.001
Mixture of Gaussian with LDA	97.7%	97.3%	50.00	0.0005
k-NN	100%	91.25%	0.0	0.18
k-NN with LDA	100%	94.75%	0.0	0.01
SVM	100%	94.95%	7,020.00	0.06
SVM with LDA	97.54%	96.60%	1,850.00	0.02
HDR	100%	93.80%	3,901.00	0.10

Results

TABLE II
PERFORMANCE OF BAGGING

Component Classifier	Accuracy		Time (sec)	
	Training	Testing	Training	Testing (per image)
LVQ	99.2%	97.4%	460.00	0.11
LVQ with LDA	97.8%	97.4%	95.00	0.005
SVM	99.80%	95.08%	7,850.00	0.61
SVM with LDA	97.58%	96.65%	2,150.00	0.18