EE4-62
EE9-SO25

IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2013

MSc and EEE/EIE PART IV: MEng and ACGI

MACHINE LEARNING FOR COMPUTER VISION

Monday, 29 April 10:00 am

Time allowed: 3:00 hours

There are FIVE questions on this paper.

Answer FOUR questions.

All questions carry equal marks.

Any special instructions for invigilators and information for candidates are on page 1.

Examiners responsible First Marker(s) : T-K. Kim
Second Marker(s) : C. Ling

1. (Gaussian Mixture Model and EM)

The Gaussian distribution is

$$N(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\},$$

the log of data probability of $\mathbf{x} = (x_1, \ldots, x_N)^T$ is

$$\ln p(\mathbf{x}|\mu,\sigma^2) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi).$$

Denote z as the 1-of-K representation such that $z_k \in \{0,1\}$ and $\sum_k z_k = 1$. We define

$$p(z_k = 1) = \pi_k, \quad \text{where} \quad 0 \le \pi_k \le 1, \quad \sum_{k=1}^{K}\pi_k = 1.$$

a) Given $p(z) = \prod_{k=1}^{K}\pi_k^{z_k}$ and $p(x|z) = \prod_{k=1}^{K}N(x|\mu_k,\sigma_k)^{z_k}$, show that the marginal distribution $p(x)$ is a linear superposition of Gaussian distributions. [5]

b) Explain the maximum-likelihood solution and show the expression of the log likelihood function of GMM for given a data set, $\mathbf{x} = (x_1, \ldots, x_N)^T$. See Figure 1.1. [5]

c) In the maximum-likelihood solution, explain how to calculate μ_k, and then derive the formulation for μ_k. For the derivation, define $\gamma(z_{nk})$ as

$$\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k,\sigma_k)}{\sum_{j=1}^{K}\pi_j N(x_n|\mu_j,\sigma_j)}.$$

[8]

d) Explain the EM algorithm using the variables, γ, μ and σ, E-step and M-step, and the termination of the loop. No formulation derivations are needed. [7]

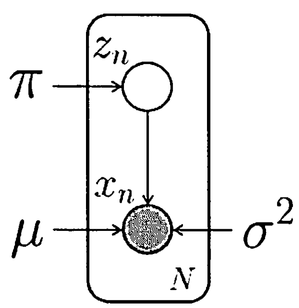


Figure 1.1 Probablistic model of GMM.

2. (Boosting)

A boosting classifier is given as a linear combination of base classifiers such that

$$f_m(x) = \frac{1}{2} \sum_{l=1}^{m} \alpha_l y_l(x).$$

a) Given a classifier f_m and a data set (x_n, t_n), where $n = 1, \ldots, N$ and t_n is the data label, define the exponential error function E.

[5]

b) Show that the error function is given in the form of $E = \sum_{n=1}^{N} w_n^{(m)} \exp\left(-\frac{1}{2} t_n \alpha_m y_m(x_n)\right)$. What is the equation for $w_n^{(m)}$? Explain how $w_n^{(m)}$ can be treated as constants?

[7]

c) Discuss the pros and cons of using the exponential error function.

[5]

d) Explain the AdaBoost algorithm by mentioning the steps of initialising data weights, finding the best weak-classifier $y_m(x)$ and re-weighting data samples in the rounds of boosting.

[8]

3. (Random Forest and Committee Machine)

a) Explain the Bagging (bootstrap aggregation) process.

[7]

b) The average error made by models acting individually is

$$E_{av} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}\left[\varepsilon_m(x)^2\right],$$

where the output of each model $y_m(x) = h(x) + \varepsilon_m(x)$, and $h(x)$ and $\varepsilon_m(x)$ denote the true value and the error of each model respectively.

Calculate the expected error E_{com} of the committee machine

$$y_{com}(x) = \frac{1}{M}\sum_{m=1}^{M} y_m(x).$$

Show that $E_{com} = \frac{1}{M}E_{av}$, and explain what condition is required to hold the equation.

[10]

c) In the RF (Random Forest) node splitting as shown in Figure 3.1, we maximise the information gain $\Delta E = -\frac{|I_l|}{|I_n|}E(I_l) - \frac{|I_r|}{|I_n|}E(I_r)$. Explain what is the meaning of the function E and ΔE, and what are the desired distributions of the two children nodes.

[8]

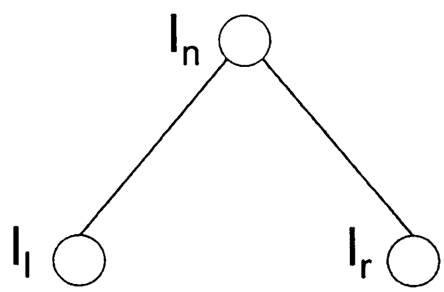


Figure 3.1 RF node splitting.

4. (Sparse Kernel Machine and Bag of Words)

a) Explain how to build a visual dictionary and how to represent an image by the Bag of Words.

[5]

b) Explain the maximum margin classifier using the function

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\mathbf{w}\|}\min_{n}\left[t_n(\mathbf{w}^T\phi(\mathbf{x}_n)+b)\right]\right\}.$$

Address the meaning of $\mathbf{w}, b, t_n$, the kernel function ϕ, and the margin (i.e. the minimum perpendicular distance).

[7]

c) Show the constrained optimisation problem, $L(\mathbf{w},b,a)$, equivalent to the max margin problem above, with the Lagrange multipliers $a_n > 0$.

[7]

d) Explain the two approaches for multi-class classification using binary SVMs (support vector machines): the one-versus-the-rest and one-versus-one approach.

[6]

5. (PCA: maximum-variance formulation)

Given a data set $\{x_n\}$, $n = 1, \ldots, N$, $x_n \in R^D$, the goal is to project the data onto a space of dimension $M << D$ while maximising the projected data variance. Assume $M = 1$. The projection is $\tilde{x}_n = u_1^T x_n$, where $u_1^T u_1 = 1$. See Figure 5.1

a) Define the data covariance matrix S, and then show the mean and the variance of the projected data using S.

[5]

b) Show the objective function of PCA using a Lagrange multiplier λ_1.

[5]

c) Show that the solution of u_1 is the eigenvector of S.

[5]

d) Show that the maximum variance is obtained as the eigenvalue.

[5]

e) Explain the optimal linear projections for the general case of an M dimensional subspace.

[5]

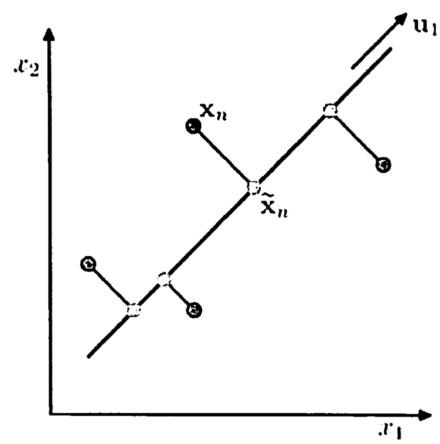


Figure 5.1 Maximum-variance formulation of PCA.

1a) The marginal distribution of x is

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

which is as a linear superposition of Gaussians.

1b) We find the three parameters that maximise the log likelihood function.

The log of the likelihood function is

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k) \right\}$$

1c) Setting the derivatives of $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$ with respect to μ_k to zero, we obtain

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\mu_j, \Sigma_j)}}_{\gamma(z_{nk})} \Sigma_k(\mathbf{x}_n - \mu_k)$$

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})\mathbf{x}_n, \quad N_k = \sum_{n=1}^{N} \gamma(z_{nk}).$$

1d)

1. Initialise the means μ_k, covariances Σ_k and mixing coefficients π_k, and evaluate the initial value of the log likelihood.

2. **E** step: Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\mu_j, \Sigma_j)}$$

3. **M** step: RE-estimate the parameters using the current responsibilities

$$\mu_k^{new} = \frac{1}{N_k}\sum_{n=1}^{N} \gamma(z_{nk})\mathbf{x}_n \qquad \Sigma_k^{new} = \frac{1}{N_k}\sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \mu_k^{new})(\mathbf{x}_n - \mu_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}, \text{ where } N_k = \sum_{n=1}^{N} \gamma(z_{nk}).$$

4. Evaluate the log likelihood

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k) \right\}$$

and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied, return to step 2.

2a)

$$E = \sum_{n=1}^{N} \exp\{-t_n f_m(\mathbf{x}_n)\}$$

2b)

$$= \sum_{n=1}^{N} \exp\left\{-t_n f_{m-1}(\mathbf{x}_n) - \frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\right\} = \sum_{n=1}^{N} w_n^{(m)} \exp\left\{-\frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\right\}$$

where $w_n^{(m)} = \exp\{-t_n f_{m-1}(\mathbf{x}_n)\}$ are constants.

- *Sequential minimisation:* suppose that the base classifiers $y_1(\mathbf{x}), \ldots, y_{m-1}(\mathbf{x})$ and their coefficients $\alpha_1, \ldots, \alpha_{m-1}$ are fixed, and we minimise only w.r.t. α_m and $y_m(\mathbf{x})$. We can write the error function by

2c)

- *Pros:* it leads to simple derivations of Adaboost algorithms.
- *Cons:* it penalises large negative values. It is much less robust to outliers or misclassified data points.

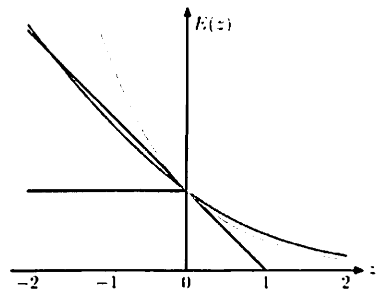


Figure 1: The exponential (green), rescaled cross entropy (red), hinge (blue) and misclassification (black) error functions.

2d)

1. Initialise the data weights $\{w_n\}$ by $w_n^{(1)} = 1/N$ for $n = 1, \ldots, N$.

2. For $m = 1, \ldots, M$

(a) Learn a classifier $y_m(\mathbf{x})$ that minimises the weighted error

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}) \neq t_n)$$

where I is the impulse function which is 1 when $y_m(\mathbf{x}) \neq t_n$.

(b) Evaluate

$$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}} \qquad \alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\}$$

(c) Update the data weighting coefficients

$$w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m I(y_m(\mathbf{x}) \neq t_n)\}$$

3. Make predictions using the final model by

$$Y_M(\mathbf{x}) = \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right)$$

3a) • Bootstrapping data sets: Given a standard training set D of size n, bagging generates m new training sets D_i, each of size $n' = n$, by sampling examples from D uniformly and with replacement. By sampling with replacement, it is likely that some examples will be repeated in each D_i. If $n' = n$, then for large n the set D_i is expected to have 63.2% of the unique examples of D, the rest being duplicates.

3b) • The expected error of the committee is

$$E_{com} = \mathbb{E}\left[\left\{\frac{1}{M}\sum_{m=1}^{M} y_m(\mathbf{x}) - h(\mathbf{x})\right\}^2\right] = \mathbb{E}\left[\left\{\frac{1}{M}\sum_{m=1}^{M} \epsilon_m(\mathbf{x})\right\}^2\right]$$

• If we assume

$$\mathbb{E}[\epsilon_m(\mathbf{x})\epsilon_l(\mathbf{x})] = 0, \quad m \neq l$$

then we obtain

$$E_{com} = \frac{1}{M}E_{av}$$

$$E_{com} = E\left(\left(\frac{1}{M}\Sigma y - h\right)^2\right) = E\left(\left(\frac{1}{M}(\Sigma y - Mh)\right)^2\right)$$

$$= E\left(\left(\frac{1}{M}\left(\sum^{M}(y-h)\right)\right)^2\right) = E\left(\left(\frac{1}{M}\Sigma\varepsilon\right)^2\right)$$

$$= E\left(\frac{1}{M^2}(\varepsilon_1 + \varepsilon_2 + \dots)^2\right) \qquad E(\varepsilon_i \varepsilon_j) = 0 \text{ if } i \neq j$$

$$= E\left(\frac{1}{M^2}\left(\varepsilon_1^2 + \varepsilon_2^2 + \cancelto{0}{\varepsilon_1\varepsilon_2 \cdot 2} + \dots\right)\right)$$

$$= \frac{1}{M} \underline{E_{av}}$$

3c) E measures the entropy (amount of info.) of the class distribution of each chide node. Uniform distribution has max entropy while peak distribution has min entropy. Thus, for classification in the final leaf nodes, we want to have both distributions peaky as possible in a node splitting.

4a)

Interest points are detected from an image

Corners, Blob detector, SIFT detector

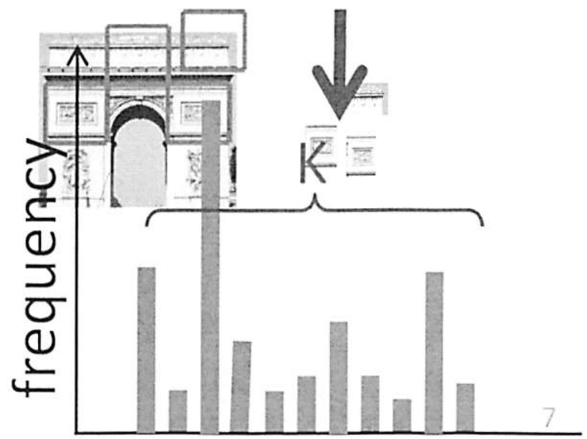
Image patches are represented by descriptor

SIFT (Scale-Invariant Feature Transform) or Raw pixel intensity

Visual words (real-valued vectors) can be compared using Euclidean distance.

These vectors are divided into groups which are similar, essentially clustered together, to form a codebook.

- Every visual word is compared with codewords and assigned to the nearest codeword.
- Histogram bins are codewords and each bin counts the number of words assigned to the codeword.

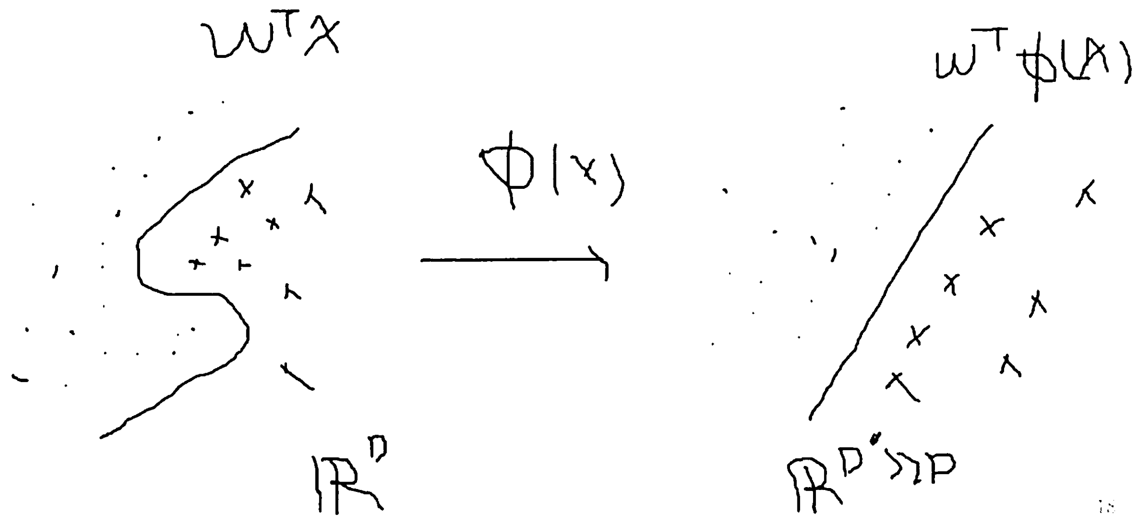


4b) • The margin is the minimum perpendicular distance, and we wish to find $\mathbf{w}$ and b that maximises the margin. The solution is found by

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{||\mathbf{w}||}\min_n[t_n(\mathbf{w}^T\phi(\mathbf{x})+b)]\right\}$$

• The perpendicular distance of a point $\mathbf{x}$ from a hyperplane $y(\mathbf{x})=0$ is $|y(\mathbf{x})|/||\mathbf{w}||$. As we assumed $t_n y(\mathbf{x}_n)>0$ for all n, the distance of a point $\mathbf{x}_n$ to the decision surface is

$$\frac{t_n y(\mathbf{x}_n)}{||\mathbf{w}||}=\frac{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)}{||\mathbf{w}||}$$



4c) Note that rescaling $\mathbf{w} \rightarrow k\mathbf{w}$ and $b \rightarrow kb$ does not change the distance from any point $\mathbf{x}_n$ to the decision surface. We can therefore set

$$t_n(\mathbf{w}^T\phi(\mathbf{x}) + b) = 1$$

for the point that is closest to the surface. In this case, all data points will satisfy

$$t_n(\mathbf{w}^T\phi(\mathbf{x}) + b) \geq 1, \quad n = 1, ..., N.$$

- The optimisation problem then simply becomes to maximise $||\mathbf{w}||^{-1}$. Equivalently, we have

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

subject to the constraints $t_n(\mathbf{w}^T\phi(\mathbf{x})+b) \geq 1$. This is a *quadratic programming* problem where we try to minimise a quadratic

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n\{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\}$$

where $\mathbf{a} = (a_1, ..., a_N)^T$. Note the minus sign in front of the Lagrange multiplier term, as we are minimising w.r.t. $\mathbf{w}, b$, and maximising w.r.t. $\mathbf{a}$.

4d) *One-versus-the-rest* approach: trains K separate SVMs, in which the k-th model $y_k(\mathbf{x})$ is trained using the data from class C_k as the positive examples and the data from the remaining $K-1$ classes as the negative examples.

The prediction for new input $\mathbf{x}$ is by

$$y(\mathbf{x}) = \max_k y_k(\mathbf{x}).$$

Problems: 1) the output values $y_k(\mathbf{x})$ for different classifiers have no appropriate scales. 2) the training sets are imbalanced.

One-versus-one approach: is to train $K(K-1)/2$ different 2-class SVMs on all possible pairs of classes, and then to classify test points according to which class has the highest number of 'votes'.

Problems: it requires more training time and evaluation time.

5a) The mean is $\mathbf{u}_1^T\overline{\mathbf{x}}$, where $\overline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$.

The variance is given by

$$\frac{1}{N}\sum_{n=1}^{N}\{\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\overline{\mathbf{x}}\}^2 = \mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$$

where $\mathbf{S}$ is the data covariance matrix defined as

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}})(\mathbf{x}_n - \overline{\mathbf{x}})^T.$$

5b) • We now maximise the projected variance $\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$ with respect to $\mathbf{u}_1$ with the normalisation condition $\mathbf{u}_1^T\mathbf{u}_1 = 1$. Lagrange multiplier formulation becomes

5c)

$$\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 + \lambda_1(1 - \mathbf{u}_1^T\mathbf{u}_1).$$

By setting the derivative with respect to $\mathbf{u}_1$ to zeros, we obtain

$$\mathbf{S}\mathbf{u}_1 = \lambda_1\mathbf{u}_1$$

5d)

thus, $\mathbf{u}_1$ is an eigenvector of $\mathbf{S}$. By multiplying $\mathbf{u}_1^T$, the variance is obtained by

$$\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 = \lambda_1.$$

5e)

- If we consider the general case of an M dimensional subspace, the optimal linear projection is defined by the M eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M$ of the data covariance matrix $\mathbf{S}$ corresponding to the M largest eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_M$.