Solutions to 4F10 Pattern Processing, 2017

1. *Bayes' Decision rule and generative models*

(a) Bayes' decision rule states

$$\text{Decide } \underset{\omega_j}{\arg\max} \; \{P(\omega_j|\boldsymbol{x})\}$$

which can be expressed for the generative classifiers here as

$$\text{Decide } \underset{\omega_j}{\arg\max} \; \{p(\boldsymbol{x}|\omega_j)P(\omega_j)\}$$

Generative models are minimum error classifiers is if there is

- infinite training data
- correct models (likelihood and priors)
- appropriate training algorithm

For most practical tasks none of these are true. [15%]

(b)(i) As the covariance matrices are diagonal each dimension can be treated separately. Thus need to maximise

$$\sum_{i=1}^{n} -\frac{y_i}{2}\left(\log(\sigma^2) + \frac{(x_i-\mu_1)^2}{\sigma^2}\right) - \frac{(1-y_i)}{2}\left(\log(\sigma^2) + \frac{(x_i-\mu_2)^2}{\sigma^2}\right)$$

Differentiating with respect to σ^2 yields

$$-\frac{n}{2\sigma^2} + \frac{1}{2}\sum_{i=1}^{n}\left(y_i\frac{(x_i-\mu_1)^2}{\sigma^4} + (1-y_i)\frac{(x_i-\mu_2)^2}{\sigma^4}\right)$$

Equating this to zero yields

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\left(y_i(x_i-\mu_1)^2 + (1-y_i)(x_i-\mu_2)^2\right)$$

[20%]

(b)(ii) The posterior for class ω_1 can be written as

$$\begin{aligned} P(\omega_1|\mathbf{x}) &= \frac{P(\omega_1)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})}{P(\omega_1)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}) + P(\omega_2)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})} \\ &= \frac{1}{1+\frac{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})}{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})}} \end{aligned}$$

It is possible to write

$$\begin{aligned} \frac{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})}{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})} &= \exp\left((\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)'\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}(\boldsymbol{\mu}_2'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1)\right) \\ &= \exp(-\mathbf{w}'\mathbf{x} + a) \end{aligned}$$

Thus

$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)' \\ a &= -\frac{1}{2}(\boldsymbol{\mu}_2'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1) \\ \boldsymbol{\phi}(\mathbf{x}) &= \mathbf{x} \end{aligned}$$

[25%]

(c) Exactly the same form as before except the ratio will now be

$$\frac{\mathcal{N}(\mathbf{x};\mathbf{0},\boldsymbol{\Sigma}_2)}{\mathcal{N}(\mathbf{x};\mathbf{0},\boldsymbol{\Sigma}_1)} = \exp\left(-\frac{1}{2}\left(\mathbf{x}'\boldsymbol{\Sigma}_2^{-1}\mathbf{x} - \mathbf{x}'\boldsymbol{\Sigma}_1^{-1}\mathbf{x} + \log(|\boldsymbol{\Sigma}_2|) - \log(|\boldsymbol{\Sigma}_1|)\right)\right)$$

As the covariance matrices are both diagonal it is possible to write

$$\begin{aligned}\frac{1}{2}\left(\mathbf{x}'\boldsymbol{\Sigma}_2^{-1}\mathbf{x} - \mathbf{x}'\boldsymbol{\Sigma}_1^{-1}\mathbf{x}\right) &= \frac{1}{2}\sum_{i=1}^{d} x_i^2\left(\frac{1}{\sigma_{2i}^2} - \frac{1}{\sigma_{1i}^2}\right)\\ &= \mathbf{w}'\boldsymbol{\phi}(\mathbf{x})\end{aligned}$$

where

$$\begin{aligned}\mathbf{w} &= \frac{1}{2}\left[\begin{array}{ccc}(\frac{1}{\sigma_{21}^2} - \frac{1}{\sigma_{11}^2}) & \cdots & (\frac{1}{\sigma_{2d}^2} - \frac{1}{\sigma_{1d}^2})\end{array}\right]'\\ a &= -\frac{1}{2}\log(|\boldsymbol{\Sigma}_2|) - \log(|\boldsymbol{\Sigma}_1|)\\ \boldsymbol{\phi}(\mathbf{x}) &= \left[\begin{array}{ccc}x_1^2 & \ldots & x_d^2\end{array}\right]'\end{aligned}$$

[25%]

(d) The decision boundary in (b) is a linear decision boundary. The form in part (c) is quadratic and results in an ellipsis centred around the origin. [15%]

Comment: This question examined Bayes' decision rule and the associated class posterior probabilities. The most popular question and generally well answered. The main issue was that some candidates did not use the fact that the covariance matrices were diagonal, simplifying the algebra.

2. *Gaussian Mixture Models and Speaker Identification*

(a) The log-likelihood of the data using just a single GMM can be expressed as

$$p(\boldsymbol{x}, \ldots, \boldsymbol{x}_N | \texttt{GMM}_k) = \sum_{i=1}^{N} \log(\sum_{m=1}^{M} c_m \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_m^{(k)}, \boldsymbol{I}))$$

[10%]

(b)(i) Expectation-maximisation is an iterative approach to estimate the model parameters. There are two distinct steps

(a) *expectation* given the current model parameters get the expected values from the auxiliary function;

(b) *maximisation* given the statistics obtain the optimal value of the auxiliary function.

The expression is guaranteed to find a local maximum. In terms of the auxiliary function

- ω_m is the component of the GMM of interest.
- z is a constant scalar.

[20%]

(b)(ii) The log-likelihood can be written as

$$\log(p(\mathbf{x}_i|\omega_m, \hat{\lambda})) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^{d}\left(x_{ij} - \sum_{k=1}^{K}\hat{\lambda}_k\mu_{mj}^{(k)}\right)'\left(x_{ij} - \sum_{k=1}^{K}\hat{\lambda}_k\mu_{mj}^{(k)}\right)$$

It is possible to write

$$\sum_{k=1}^{K}\hat{\lambda}_k\mu_{mj}^{(k)} = \boldsymbol{m}'_{mj}\hat{\boldsymbol{\lambda}}$$

where

$$\boldsymbol{m}'_{mj} = \left[\begin{array}{ccc} \mu_{mj}^{(1)} & \ldots & \mu_{mj}^{(K)} \end{array}\right]$$

It is then possible to write

$$\log(p(\mathbf{x}_i|\omega_m, \hat{\lambda})) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^{d}\left(x_{ij}^2 - 2x_{ij}\boldsymbol{m}'_{mj}\hat{\boldsymbol{\lambda}} + \hat{\boldsymbol{\lambda}}'\boldsymbol{m}'_{mj}\boldsymbol{m}_{mj}\hat{\boldsymbol{\lambda}}\right)$$

Substituting this expression into the auxiliary function yields the required expression where

$$\begin{aligned}
e &= z - \frac{Nd}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{d}x_{ij}^2 \\
\mathbf{b} &= \sum_{i=1}^{N}\sum_{m=1}^{M}P(\omega_m|\mathbf{x}_i, \boldsymbol{\lambda})\sum_{j=1}^{d}x_{ij}\boldsymbol{m}_{mj} \\
\mathbf{C} &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{m=1}^{M}P(\omega_m|\mathbf{x}_i, \boldsymbol{\lambda})\sum_{j=1}^{d}\boldsymbol{m}_{mj}\boldsymbol{m}'_{mj}
\end{aligned}$$

[35%]

(b)(iii) The estimate can be expressed as

$$\hat{\boldsymbol{\lambda}} = -\frac{1}{2}\mathbf{C}^{-1}\mathbf{b}$$

The only concern for this expression is whether $\mathbf{C}$ is invertible. At a minimum this requires that the number of components M is greater than K. [15%]

(c) Effectively the vector $\boldsymbol{\lambda}$ is being used as a sequence kernel to map the variable length set of observations (N may vary per speaker). After the vector has been computed for each of the speakers there are two distinct ways of using this (only one needs to be discussed). For both schemes at enrollment stage the vector is estimated for each speaker, $\boldsymbol{\lambda}^{(s)}$. Then either

- for the test data estimate a new $\boldsymbol{\lambda}$. Then compute the distance to each of the enrolment speakers;
- compute the likelihood of the test data for each of the $\boldsymbol{\lambda}^{(s)}$, select the speaker with the highest likelihood.

[20%]

Comment: A question examining Expectation Maximisation (EM) and its application to parameter estimation. Though a popular question this was generally poorly answered. Many candidates unable to derive the form of auxiliary function given.

3. *Support Vector Machines and Multi-Class Classification*

(a)(i) One of the two classes must be assigned for class 1 $\tilde{y}_i = 1$ and for class 2 $\tilde{y}_i = -1$. All points must lie beyond the margin. In this case all samples must satisfy

$$\tilde{y}_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}) + b) \geq 1$$

[15%]

(a)(ii) Points to be mentioned:

(a) The values of the Lagrange multipliers are found by maximising the margin.

(b) For each pair one is selected as being class 1, the second as class 2. The SVM for this pair is then trained. This allows $\alpha_i^{(pq)}$ to be found where

$$\alpha_i^{(pq)} = \tilde{y}_i \alpha_i$$

(c) For all training examples that don't belong to ω_p and ω_q $\alpha_i^{(pq)} = 0$. Effectively this will not be support vectors.

(d) The values of $b^{(pq)}$ are found in the standard fashion from the SVM. Training examples with non-zero support vectors are used and then points that satisfy

$$\tilde{y}_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}) + b) = 1$$

[20%]

(a)(iii) Two approaches mentioned in lectures, majority voting and use of adaptive DAG classification. The simplest is majority voting, discussed here.

- Each of the $K(K-1)/2$ classifiers are run.
- The class outcome from each of these is counted as a vote for each of them. There are therefore $K(K-1)/2$ SVM classification runs.
- A problem arises from "ties" where the votes for two or more of the classes are the same. For a tie between two classes is normally resolved using the classification result for these classes. For more than two classes it is not possible to guarantee that a clear winner will result.

[20%]

(b) (i) This is a standard form discussed in lectures, Kesler's construct, but applied to SVMs. If observation $\mathbf{x}_i$ belongs to class ω_1 for example and $j = 2$

$$\mathbf{z}_j = \begin{bmatrix} 1 \\ \boldsymbol{\phi}(\mathbf{x}_i) \\ -1 \\ -\boldsymbol{\phi}(\mathbf{x}_i) \\ \mathbf{0} \end{bmatrix}$$

All $K-1$ possible values of j must be considered. [15%]

(b)(ii) To train the SVM both positive and negative examples are required. These can be simply generated by using setting one of the training examples as if it belonged to a different class to it's actual one. It is then possible to form the Gram matrix, which will be $(K-1)m \times (K-1)m$. [15%]

(b)(iii) The computational cost of this approach is to perform the kernel operations in the extended space defined by $\tilde{\mathbf{w}}$. The issues to consider are

- single classifier but the observation space is K times as large.
- there are $K(K-1)/2$ classifications for a(iii) (majority vote)
- K for (b)(i) pick the largest

- the number of support vectors. This is expected to be larger for the single classifier than each of the individual ones. [15%]

Comment: This questions examined the students' knowledge of Support Vector Machines (SVMs), and how they can be extended to handling multiple classes. A number of candidates failed to understand the difference between the binary case in part (a) and the multi-class case in part (b).

4. *Parameter Optimisation and Parameter Pruning*

(a)(i) The values are

$$\begin{aligned}\mathbf{b} &= \nabla E(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}^{(\tau)}} \\ \mathbf{A} &= \nabla^2 E(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}^{(\tau)}}\end{aligned}$$

the gradient and the Hessian respectively at the current model parameters. [10%]

(a)(ii) Letting

$$\boldsymbol{\Delta\theta}^{(\tau)} = (\boldsymbol{\theta} - \boldsymbol{\theta}^{(\tau)})$$

gives the following differential expression

$$\frac{\partial}{\partial \boldsymbol{\Delta\theta}^{(\tau)}} E(\boldsymbol{\theta}) = \mathbf{b} + \mathbf{A}\boldsymbol{\Delta\theta}^{(\tau)})$$

Equating this to zero gives

$$\boldsymbol{\Delta\theta}^{(\tau)} = -\mathbf{A}^{-1}\mathbf{b}$$

Thus the value is given by

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(\tau)} - \mathbf{A}^{-1}\mathbf{b}$$

[20%]

(b)(i) At the local minimum it is known that $\mathbf{b} = \mathbf{0}$. Thus the cost function becomes

$$E(\boldsymbol{\theta}) \approx E(\boldsymbol{\theta}^{\star}) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^{\star})'\mathbf{H}(\boldsymbol{\theta} - \boldsymbol{\theta}^{\star}) = E(\boldsymbol{\theta}^{\star}) + \Delta\boldsymbol{\theta}'\mathbf{H}\Delta\boldsymbol{\theta}$$

The constraint can be expressed as

$$\Delta\theta_q + \theta_q = \mathbf{e}_q'\Delta\boldsymbol{\theta} + \theta_q = 0$$

where $\mathbf{e}_q$ selects element q of the vector. The Lagrangian can then be written as

$$l(\boldsymbol{\theta}) = E(\boldsymbol{\theta}^{\star}) + \frac{1}{2}\Delta\boldsymbol{\theta}'\mathbf{H}\Delta\boldsymbol{\theta} + \lambda(\mathbf{e}_q'\Delta\boldsymbol{\theta} + \theta_q)$$

Differentiating and equating to zero, yields

$$\begin{aligned}\mathbf{H}\Delta\boldsymbol{\theta} + \lambda\mathbf{e}_q &= \mathbf{0} \\ \mathbf{e}_q'\Delta\boldsymbol{\theta} + \theta_q &= 0\end{aligned}$$

From the first equation

$$\Delta\boldsymbol{\theta} = -\lambda\mathbf{H}^{-1}\mathbf{e}_q$$

Need to find λ. Multiply the above expression by $\mathbf{e}_q'$ and combining with the second equation

$$\begin{aligned}\mathbf{e}_q'\Delta\boldsymbol{\theta} &= -\theta_q \\ &= -\lambda\mathbf{e}_q'\mathbf{H}^{-1}\mathbf{e}_q \\ &= -\lambda[\mathbf{H}^{-1}]_{qq}\end{aligned}$$

Thus

$$\lambda = \frac{\theta_q}{[\mathbf{H}^{-1}]_{qq}}$$

Combining these expressions together yields

$$\Delta\boldsymbol{\theta} = -\frac{\theta_q}{[\mathbf{H}^{-1}]_{qq}}\mathbf{H}^{-1}\mathbf{e}_q$$

[30%]

(b)ii) The change in the score is given by

$$\begin{aligned} E(\boldsymbol{\theta}^\star) - E(\hat{\boldsymbol{\theta}}) &= \frac{1}{2}\Delta\boldsymbol{\theta}'\mathbf{H}\Delta\boldsymbol{\theta} \\ &= \frac{1}{2}\frac{\theta_q^2}{[\mathbf{H}^{-1}]_{qq}} \end{aligned}$$

[15%]

(b)(iii) An iterative approach is used where the links that yield the smallest increase in the auxiliary function are removed. [10%]

(b)(iv) When the **H** is restricted to be diagonal, the change in the parameters simply becomes $-\theta_q$ (all parameters are assumed to be independent). It is now highly efficient as the Hessian inversion is not altered by the pruning process. [15%]

Comment: This question was based around second-order optimisation approaches for neural networks. The first part of the question was well done. However the constrained optimisation part was poorly answered, with few candidates being able to set-up the Lagrangian correctly.

5. *Neural Network Configuration and Activation Functions*

(a) The main concern is the generalisation of the network. It is important that the network parameters can be robustly estimated. For the system described the numbers of parameters, N, is

$$N = (d+1) \times N_1 + K \times (N_L + 1) + \sum_{l=1}^{L-1} (N_l + 1) \times N_{l+1}$$

Empirically it has been found that having multiple layers (a deep network) is better than having a single wide layer. [15%]

(b)(i) [This is a slightly modified version of an examples paper question] Need to compute the first and second moments. First moment given by

$$\int_{-\infty}^{\infty} \phi(x)p(x)dx = \beta \int_{-\infty}^{0} xp(x)dx + \alpha \int_{0}^{\infty} xp(x)dx = \alpha \int_{0}^{\infty} xp(x)dx - \beta \int_{0}^{\infty} xp(x)dx$$

It is possible to show that when $p(x) = \mathcal{N}(x; 0, \sigma^2)$

$$\int_{0}^{\infty} xp(x)dx = \frac{\sigma}{2}\sqrt{\frac{2}{\pi}}$$

Hence

$$\int_{-\infty}^{\infty} \phi(x)p(x)dx = (\alpha - \beta)\frac{\sigma}{2}\sqrt{\frac{2}{\pi}} = (\alpha - \beta)\sigma\sqrt{\frac{1}{2\pi}}$$

and the second moment

$$\int_{-\infty}^{\infty} (\phi(x))^2 p(x)dx = \int_{-\infty}^{0} \alpha^2 x^2 p(x)dx + \int_{0}^{\infty} x^2\phi(x)p(x)dx = (\alpha^2 + \beta^2)\sigma^2/2$$

So the total variance on the output is

$$\hat{\sigma}^2 = (1+\alpha^2)\sigma^2/2 - (1-\alpha)^2\frac{\sigma^2}{2\pi} = \left[(1+\alpha^2)/2 - (1-\alpha)^2\frac{1}{2\pi}\right]\sigma^2$$

[30%]

(b)(ii) The simplest approach is to ensure that the output variance matches the input variances for the initialisation (as discussed in lectures). Thus if there are N_l nodes entering the an activation function for layer $l+1$, the resulting variance of the input of the activation function would be (assume that the observation variance has been set to one)

$$(N_l + 1)\sigma_{\mathtt{w}}^2$$

where $\sigma_{\mathtt{w}}^2$ is the variance of the weight initialisation. Thus the variance on the output will be

$$\left[(1+\alpha^2)/2 - (1-\alpha)^2\frac{1}{2\pi}\right](N_l + 1)\sigma_{\mathtt{w}}^2$$

This function has an added complexity as the mean is non-zero. If the network is deep this could result in a large offset for some layers. To address this the weights of the bias need to be set appropriately as well, the mean is set to

$$-\alpha \int_{0}^{\infty} xp(x)dx + \beta \int_{0}^{\infty} xp(x)dx$$

[25%]

(c)(i) Setting the two values to be the same results in the activation function being linear. Thus there is no advantage in having deep networks, and hidden layers with more units than d. The mean of the output is now unaltered, so the bias can be zero mean. The network still needs to ensure that the variance going to the softmax is sensible - operating within a sensible range. [15%]

(c)(ii) α_0 has no impact on the number of layers, as everything is linear. The initialisation will need to be considered due to the soft-max. The value will not impact the overall performance of the system. [10%]

(d) Training the parameters has the potential to improve performance as the impact of α and β cannot be modelled by training the weights. [10%]

Comment: This question examined the candidates knowledge of network topologies and activation functions. This was the least popular question despite a similar question being asked in an examples paper.