4F10 Deep Learning and Structured Data, 2023

1. *Expectation Maximisation and Gaussian Mixture Models*

(a) Log-likelihood of the training data is

$$\begin{aligned}\log(p(x_1,\ldots,x_N|\theta)) &= \sum_{i=1}^{N}\log\left(\sum_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}\left(\prod_{i=1}^{n}a_{\theta_{i-1}\theta_i}\right)\sum_{m=1}^{M}c_m\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)\right)\\ &= \sum_{i=1}^{N}\log\left(\sum_{m=1}^{M}c_m\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)\right)\end{aligned}$$

Acceptable to say no additional information. Possible to also say the only additional information that the HMM can model is a minimum duration by having left-right topology. [15%]

(b)(i) Substituting in the expression for the likelihood to the auxiliary function

$$\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}\sum_{j=1}^{J}\sum_{m=1}^{M}P(\mathbf{s}_j,\omega_m|x_i,\boldsymbol{\theta})\log\left(\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)\right)$$

Differentiate this with respect to $\boldsymbol{\mu}_q$ gives

$$\frac{\partial\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}})}{\partial\hat{\boldsymbol{\mu}}_q} = \sum_{i=1}^{n}\sum_{j=1}^{J}P(\mathbf{s}_j,\omega_q|x_i,\boldsymbol{\theta})\left[\hat{\boldsymbol{\Sigma}}_q^{-1}(\mathbf{x}_i-\hat{\boldsymbol{\mu}}_q)\right]$$

Equating to zero (and setting variable to m) gives

$$\hat{\boldsymbol{\mu}}_m = \frac{\sum_{i=1}^{N}\sum_{j=1}^{J}P(\mathbf{s}_j,\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\mathbf{x}_i}{\sum_{i=1}^{N}\sum_{j=1}^{J}P(\mathbf{s}_j,\omega_m|\mathbf{x}_i,\boldsymbol{\theta})}$$

This can be further simplified by writing

$$P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta}) = \sum_{j=1}^{J}P(\mathbf{s}_j,\omega_m|\mathbf{x}_i,\boldsymbol{\theta})$$

This form is used in the following equations. [20%]

(b)(ii) As the covariance matrix is diagonal, possible to write auxiliary function as

$$\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}\sum_{m=1}^{M}P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\sum_{j=1}^{d}\left(-\log(\sqrt{2\pi}\hat{\sigma}) - \frac{1}{2}\frac{(x_{ij}-\hat{\mu}_{mj})^2}{\hat{\sigma}_j^2}\right)$$

Differentiating this wrt to $\hat{\sigma}_j$ yields

$$\sum_{i=1}^{n}\sum_{m=1}^{M} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\left(-\frac{1}{\hat{\sigma}_j}+\frac{(x_{ij}-\hat{\mu}_{mj})^2}{\hat{\sigma}_j^3}\right)$$

Equating to zero yields

$$\hat{\sigma}_j^2=\frac{\sum_{i=1}^{n}\sum_{m=1}^{M} P(\omega_m|x_i,\boldsymbol{\theta})(x_{ij}-\hat{\mu}_{mj})^2}{\sum_{i=1}^{n}\sum_{m=1}^{M} P(\omega_m|x_i,\boldsymbol{\theta})}$$

[20%]

(c)(i) As the covariance matrix is diagonal, possible to write auxiliary function as

$$\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}})=\sum_{i=1}^{n}\sum_{m=1}^{M} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\sum_{j=1}^{d}\left(-\log(\sqrt{2\pi\hat{\alpha}_m}\hat{\sigma}_j)-\frac{1}{2}\frac{(x_{ij}-\hat{\mu}_j)^2}{\hat{\alpha}_m\hat{\sigma}_j^2}\right)$$

Differentiating wrt to a_m yields

$$\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\sum_{j=1}^{d}\left(-\frac{1}{2\hat{\alpha}_m}+\frac{1}{2}\frac{(x_{ij}-\hat{\mu}_j)^2}{\hat{\alpha}_m^2\hat{\sigma}_j^2}\right)$$

Equating to zero yields

$$\hat{\alpha}_m=\frac{\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\sum_{j=1}^{d}(x_{ij}-\hat{\mu}_j)^2/\hat{\sigma}_j}{d\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})}$$

Unfortunately $\hat{\alpha}_m$ is a function of $\hat{\sigma}$. It is therefore necessary to interleave the updates - this is a GEM update (not discussed in lectures). [30%]

(c)(ii) Discussion should include:

- model in (b) allows multi-modal data to be modelled, (c) is unimodal;
- model in (b) allows non-symmetric data to be modelled, (c) is symmetric;
- (c) is a method to model symmetric data that is non-Gaussian efficiently;
- (c) has significantly fewer parameters (depending on d);
- computational contrast (possible to efficiently compute tied variance systems).

[15%]

2. *Deep Learning and Sequence models*

(a)(i) The basic steps that need to be described are:

i. take a 1-of-K encoding of the French words and embed them into embedding $\mathbf{h}_1, \ldots, \mathbf{h}_L$. Any form of (reasonable) embedding structure is acceptable;

ii. apply an attention mechanism over the input sequence

$$\mathbf{h} = \sum_{i=1}^{L} \alpha_i \mathbf{h}_i$$

a self attention mechanism is used, for example

$$\alpha_i = \frac{1}{Z} \exp\left(\mathbf{h}_i^{\mathsf{T}} \mathbf{A} \mathbf{h}_i\right)$$

where $\mathbf{A}$ is a trainable matrix.

iii. a classifier is then put on the output, The simplest form is

$$P(\omega_i | \omega_{1:L}^{\mathtt{f}}) = \frac{\exp(\mathbf{w}_i^{\mathsf{T}} \mathbf{h} + b_i)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^{\mathsf{T}} \mathbf{h} + b_j)}$$

where $\mathbf{w}_i$ is the weight vector associated with class ω_i and b_i the bias. [25%]

(a)(ii) The standard training criterion is cross-entropy

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \sum_{k=1}^{K} \delta(\omega_k, \omega^{(i)}) P(\omega_k | \omega_{1:L}^{\mathtt{f}(i)})$$

[15%]

(b)(i) Given the encoded French word sequence, $\mathbf{h}_1, \ldots, \mathbf{h}_L$ the following processes should be described

i. encode the back-history of generated German words to predict the i-th word

$$\hat{\omega}_{1:i-1}^{\mathtt{g}} \rightarrow \mathbf{h}_i^{\mathtt{g}}$$

any reasonable form of encoding is acceptable.

ii. compute the attention to predict the i-th word

$$\mathbf{c}_i = \sum_{j=1}^{L} \alpha_j \mathbf{h}_j; \quad \alpha_j = \frac{1}{Z} \exp\left(\mathbf{h}_i^{\mathtt{g}\mathsf{T}} \mathbf{A} \mathbf{h}_j\right)$$

iii. combine the context information with the back-history

$$P(\omega_k | \hat{\omega}_{1:i-1}^{\mathtt{g}}, \omega_{1:L}^{\mathtt{f}}) = \frac{\exp(\mathbf{w}_k^{\mathsf{T}} \hat{\mathbf{h}}_i + b_i)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^{\mathsf{T}} \hat{\mathbf{h}}_i + b_j)}$$

where $\hat{\mathbf{h}}_i = [\mathbf{c}_i^{\mathsf{T}}, \mathbf{h}_i^{\mathtt{g}\mathsf{T}}]^{\mathsf{T}}$, and ω_k is a word from the German vocabulary.

iv. select the most probable class from $P(\omega_k|\hat{\omega}^{\mathtt{g}}_{1:i-1}, \omega^{\mathtt{f}}_{1:L})$ and use that as $\hat{\omega}^{\mathtt{g}}_i$. [40%]

(b)(ii) The simplest approach is to use cross-entropy on the output

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N}\sum_{j=1}^{J}\sum_{k=1}^{V}\delta(\omega_k, \omega_j^{\mathtt{g}(i)})P(\omega_k|\omega^{\mathtt{g}(i)}_{1:j-1}, \omega^{\mathtt{f}(i)}_{1:L})$$

where V is the size of the German vocabulary. This is the standard *teacher-forcing approach*. The limitation is that there is a mismatch at inference time between the history used in training, the reference, and hypothesised history. The advantage of the approach is that the updates can all be done in parallel compared to using hypothesised histories that must be generated in an auto-regressive fashion. [20%]

3. *Support vector machines, margin, kernels*

(a) A support vector will have a non-zero Lagrange multiplier in the solution to the SVM dual problem. In a hard margin SVM, it is also one of the data points that is closest to the classifier's decision boundary and the output of the classifier at the support vector times the class label will be equal to 1. If you remove all data points except the support vectors, the resulting hard margin SVM classifier will be the same. [15%]

(b) i. The distance is given by

$$\frac{\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x})}{||\mathbf{w}||} = \frac{4}{\sqrt{1+4+4+1}} = \frac{4}{\sqrt{10}} \approx 1.27\,. \tag{1}$$

[15%]

ii. The margin magnitude is

$$\frac{1}{||\mathbf{w}||} = \frac{1}{\sqrt{10.00}} \approx 0.32\,. \tag{2}$$

[15%]

iii. The kernel function is given by

$$\begin{aligned} k_1(\mathbf{x}', \mathbf{x}) &= \phi(\mathbf{x}')^{\mathrm{T}}\phi(\mathbf{x}) \\ &= (x_1')^2(x_1)^2 + (x_2')^2(x_2)^2 + 2x_1'x_2'x_1x_2 + 2x_1'x_1 + 2x_2'x_2 \\ &= (x_1'x_1)^2 + (x_2'x_2)^2 + 2x_1'x_2'x_1x_2 + 2x_1'x_1 + 2x_2'x_2 \\ &= (x_1'x_1)^2 + (x_2'x_2)^2 + 2x_1'x_2'x_1x_2 + 2x_1'x_1 + 2x_2'x_2 + 1 - 1 \\ &= (x_1'x_1 + x_2'x_2)^2 + 2(x_1'x_1 + x_2'x_2) + 1 - 1 \\ &= ((\mathbf{x}')^{\mathrm{T}}\mathbf{x})^2 + 2(\mathbf{x}')^{\mathrm{T}}\mathbf{x} + 1 - 1 \\ &= (1 + \mathbf{x}^{\mathrm{T}}\mathbf{x}')^2 - 1\,. \end{aligned} \tag{3}$$

[15%]

iv. The original kernel uses an implicit feature space given by polynomials of degree 2 with no intercept. The Gaussian kernel will be preferred whenever the true decision border cannot be well described by such polynomials. [15%]

(c) i. The resulting classifier should pass through the point $[3, 4]^{\mathrm{T}}$, so its output should be zero at this point. The slope of the classifier should be -1. That is, w should be orthogonal to the vector $[1, -1]^{\mathrm{T}}$. This gives the equations

$$w_1 = w_2\,, \tag{4}$$

$$3w_1 + 4w_2 + b = 0\,, \tag{5}$$

resulting in

$$7w_1 + b = 0\,. \tag{6}$$

Furthermore, the output of the classifier at $[2,3]^{\mathrm{T}}$ should be 1:

$$2w_1 + 3w_2 + b = 1\,, \tag{7}$$

resulting in

$$5w_1 + b = 1\,. \tag{8}$$

Subtracting (6) from (8) yields

$$-2w_1 = 1 \rightarrow w_1 = w_2 = -0.5\,. \tag{9}$$

We can then infer that $b = 3.5$. [15%]

ii. We compute the output of the classifier at each point:

$$f(\mathbf{x}_1 = [1,4]^{\mathrm{T}})t_1 \quad = 1\,, \tag{10}$$
$$f(\mathbf{x}_2 = [2,3]^{\mathrm{T}})t_2 \quad = 0.92\,, \tag{11}$$
$$f(\mathbf{x}_3 = [4,5]^{\mathrm{T}})t_3 \quad = 0.36\,, \tag{12}$$
$$f(\mathbf{x}_4 = [5,6]^{\mathrm{T}})t_4 \quad = 1\,. \tag{13}$$

We, therefore, have that $\xi_1 = 0$, $\xi_2 = 0.08$, $\xi_3 = 0.64$ and $\xi_4 = 0$. From KKT conditions, we know that $\mu_n \xi_n = 0$. We, therefore, know that $\mu_2 = 0$ and $\mu_3 = 0$. We also know that $a_n = C - \mu_n$. Therefore, $a_2 = a_3 = C = 0.1$. We also know that

$$\mathbf{w} \quad = \textstyle\sum_n a_n t_n \mathbf{x}_n = a_1[1,4]^{\mathrm{T}} + 0.1[2,3]^{\mathrm{T}} + 0.1[-4,-5]^{\mathrm{T}} + a_4[-5,-6]^{\mathrm{T}} \tag{14}$$

Therefore,

$$w_1 \quad = a_1 - 5a_4 - 0.2 = -0.36\,, \tag{15}$$
$$w_2 \quad = 4a_1 - 6a_4 - 0.2 = -0.28\,, \tag{16}$$

Subtracting 4 times the first equation from the second one yields

$$14a_4 + 0.6 = 1.16 \rightarrow a_4 = (1.16 - 0.6)/14 = 0.04\,. \tag{17}$$

We can then solve for a_1 obtaining $a_1 - 5 \times 0.04 - 0.2 = -0.36 \rightarrow a_1 = 0.04$. We then have that $\mu_1 = C - a_1 = 0.1 - 0.04 = 0.06$ and $\mu_4 = C - a_4 = 0.06$. [10%]

4. *Bayes' Decision Rule and Ensembles*

(a) Bayes' decision rule for a two class problem is

$$\text{Decide} \begin{cases} \text{Class } \omega_1 & \text{if } P(\omega_1|\boldsymbol{x}) > P(\omega_2|\boldsymbol{x}); \\ \text{Class } \omega_2 & \text{Otherwise} \end{cases}$$

[10%]

(b)(i) The posterior can be expressed as, noting equal priors,

$$\begin{aligned} P(\omega_1|\boldsymbol{x}^\star) &= \frac{p(\boldsymbol{x}^\star|\omega_1)P(\omega_1)}{p(\boldsymbol{x}^\star|\omega_1)P(\omega_1) + p(\boldsymbol{x}^\star|\omega_2)P(\omega_2)} \\ &= \frac{1}{1 + \frac{p(\boldsymbol{x}^\star|\omega_2)}{p(\boldsymbol{x}^\star|\omega_1)}} \end{aligned}$$

The class-conditional PDFs are Gaussian with equal variances

$$\frac{p(\boldsymbol{x}^\star|\omega_2)}{p(\boldsymbol{x}^\star|\omega_1)} = \exp\left(\frac{1}{\alpha}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\mathsf{T}\boldsymbol{x}^\star - \frac{1}{2\alpha}(\boldsymbol{\mu}_2^\mathsf{T}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\mu}_1)\right)$$

This yields

$$P(\omega_1|\boldsymbol{x}^\star) = \frac{1}{1 + \exp\left(\frac{1}{\alpha}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\mathsf{T}\boldsymbol{x}^\star - \frac{1}{2\alpha}(\boldsymbol{\mu}_2^\mathsf{T}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\mu}_1)\right)}$$

[25%]

(b)(ii) We now need to compute, using the expression from b(i)

$$P(\omega_1|\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star) = \frac{1}{1 + \frac{p(\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star|\omega_2)}{p(\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star|\omega_1)}}$$

The measurements are independent so that

$$\begin{aligned} \frac{p(\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star|\omega_2)}{p(\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star|\omega_1)} &= \prod_{i=1}^{5} \frac{p(\boldsymbol{x}_i^\star|\omega_2)}{p(\boldsymbol{x}_i^\star|\omega_1)} \\ &= \prod_{i=1}^{5} \exp\left(\frac{1}{\alpha}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\mathsf{T}\boldsymbol{x}_i^\star - \frac{1}{2\alpha}(\boldsymbol{\mu}_2^\mathsf{T}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\mu}_1)\right) \\ &= \exp\left(\frac{1}{\alpha}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\mathsf{T}\left(\sum_{i=1}^{5}\boldsymbol{x}_i^\star\right) - \frac{5}{2\alpha}(\boldsymbol{\mu}_2^\mathsf{T}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\mu}_1)\right) \end{aligned}$$

This has the same form as Part b(i). [20%]

(c)(i) In lectures a range of approaches were described for deep ensembles:

- bagging: select random subsets of training data

- Monte-Carlo Dropout: use dropout in training and generate a ensemble from different selections of dropout
- random initialisation: use different random initialisation for th training of the network

Any approach is acceptable. [15%]

(c)(ii) The simplest approach to combine the predictions is to average them thus

$$P(\omega_1|\boldsymbol{x}^\star) = \frac{1}{M}\sum_{i=1}^{M} P(\omega_1|\boldsymbol{x}^\star, \boldsymbol{\theta}^{(i)})$$

This assumes that each of the models is approximately a draw from the parameter distribution. [10%]

(c)(iii) The simplest approach to classifier the multiple samples is to average the samples and use this averaged sample in the posterior classifier. Thus

$$P(\omega_1|\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_5^\star) \approx \frac{1}{M}\sum_{i=1}^{M} P(\omega_1|\overline{\boldsymbol{x}}^\star, \boldsymbol{\theta}^{(i)}); \quad \overline{\boldsymbol{x}}^\star = \frac{1}{5}\sum_{i=1}^{5} \boldsymbol{x}_i^\star$$

The limitation of this simple approach is that the individual classifiers were trained on single samples, not averaged samples. Thus the posteriors that come out will not necessarily be matched the test condition, and the decision boundary possibly not optimal. This could be addressed by training on ensembles of samples. [20%]