COM4509 and COM6509 Revision Worksheet (Second half of module) - Solutions

To help with the first part of the module we have prepared this exercise sheet. It covers *some* of the key topics you should ensure you spend time revising. For a few of the questions we have provided an answer, indicated by a (Q). We won't be providing answers to all these questions but you should be able to find all the answers in the lecture slides. We have highlighted some questions with [$\star$] or [$\star \star \star$]. These aren't just the ones that are most important, but are the ones that people are most likely to get wrong and we think need extra attention.

Solutions

These are selected solutions, the emphasis was to create a list of areas to stimulate your revision. You should revise more than is listed here, studying the content (lectures, labs, exercise sheets) as well as reading the recommended texts.

Q2.1

The Bernoulli distribution for a random variable Y is

$$P(Y = 1) = \pi$$
$$P(Y = 0) = 1 - \pi$$

where π is the probability of positive outcome $Y = 1$ and is the only parameter for the distribution. These 2 equations can be combined, we use y to represent either 0 or 1 giving

$$P(Y = y) = \pi^y \ (1 - \pi)^{(1-y)}.$$

We can check this by inserting $y = 1$ and $y = 0$ to recover the above equations.

Q 2.4

First we recall the important assumption that the samples are independent, this means the joint probability of all the samples $\mathbf{x}_i$ (with corresponding outcomes y_i) can be factorise as

$$L = p(\mathbf{x}_1, \cdots, \mathbf{x}_N; y_1, \cdots, y_N) = \Pi_{i=1}^{N} p(\mathbf{x}_i; y_i).$$

Please note I am not using the conditional probability here for clarity but you could write this as the probability of x given y. The log-likelihood is simply the (natural) logarithm of this expression, so the negative log-likelihood is

$$\begin{aligned} n &= -\log L \\ &= -\sum_{i=1}^{N} \log p(\mathbf{x}_i; y_i). \end{aligned}$$

where we make use of the rule that $\log(AB) = \log(A) + \log(B)$.

To optimise we now take the derivative and set this equal to 0. For logistic regression we use the probability distribution in Q2.1, taking the log of this gives

$$\begin{aligned}\log p(\mathbf{x}_i; y_i) &= \log\left(\pi(\mathbf{x}_i)^{y_i}\ (1-\pi(\mathbf{x}_i))^{(1-y_i)}\right)\\ &= \log\left(\pi(\mathbf{x}_i)^{y_i}\right) + \log\left((1-\pi(\mathbf{x}_i))^{(1-y_i)}\right)\\ &= y_i \log \pi(\mathbf{x}_i)^{y_i} + (1-y_i)\log\left((1-\pi(\mathbf{x}_i)\right)\end{aligned}$$

where we are using $\pi(\mathbf{x}_i) = \text{Sigmoid}\left(\mathbf{w}^T\mathbf{x}_i\right)$ to learn the probability that defines the distribution. Using this we can differentiate but solving the non-linear equation requires some iterative method.

Q 3.2

The adjoints for reverse mode auto-differentiation are

$$\bar{v}_i = \frac{\partial y_j}{\partial v_i}.$$

where we aim to write these in terms of adjacent intermediate variables within our computational graph. The forward modes instead uses

$$\dot{v}_i = \frac{\partial v_i}{\partial x_j},$$

so the forward works out the gradient with respect to the inputs but reverse mode works out the gradient of the output with respect to the nodes.

Q 4.3

The hidden layers in neural networks allow us to learn finer-grained features of our inputs and thus to create non-linear decision boundaries with separation in higher-dimensional spaces. They allow for automatic learning of these features but the downside is increased computation cost, interpretability and needs more data to train the additional weights.

Q 4.6

In a fully connected neural network we will need to flatten the images, therefore we will have $30 \times 45 = 1350$ inputs for the first layer. We have 5 classes so if we are using a one-hot encoding we will have 5 outputs from the network.

Q 5.2

Let's consider only one dimension as this can easily be mapped to the second. If the image is of size N, and the convolution has a filter size F, with a stride of S and padding P then the output size is

$$O = \frac{N - F + 2P}{S} + 1$$

and if this is fractional, then we will take the lowest integer. Some example, if the image is $N = 28$, with convolution $F = 5$, $P = 2$, $S = 1$ then

$$\begin{aligned}O &= \frac{28 - 5 + 2 \times 2}{1} + 1\\ &= 28\end{aligned}$$

or is $S = 2$ we will have

$$\begin{aligned} O &= \frac{28 - 5 + 2 \times 2}{2} + 1 \\ &= \frac{27}{2} + 1 \\ &= 14.5 \text{ round to } 14. \end{aligned}$$

Q 5.3

Let us assume we have the data in Q 4.6 so an image of 30 by 45 with 5 classes, let us assume they are colour (RGB). A simple suggestion would be

- Colour means 3 input channels, so start with $F = 7$, $P = 3$, $S = 1$ with 8 output channels. This returns a shape of 30 by 45 with 8 channels.
- Non-linear activation function e.g ReLU.
- For the second layer we have 8 input channels from the previous layer. We could then have $F = 5$, $P = 2$, $S = 3$ with 16 output channels. The stride will help reduce the image size as it should come out as 10 by 15 with 16 channels.
- Non-linear activation function e.g ReLU.
- A fully connected - input size $= 10 \times 15 \times 16 = 2,400$, output size 600.
- Non-linear activation function e.g ReLU
- Final fully connected layer - input size 600, output = 5.
- We could then use mean squared error to define a optimisation criterion for the network.

There is no guarantee that this will work well, a main idea is the decrease the image size while increasing output channels before applying fully connected layers to classify.

Q 6.5

The optimisation criterion that we want to **maximise** for the principal component $\mathbf{u}_1$ is

$$L(\mathbf{u}_1, \lambda_1) = \mathbf{u}_1^T \mathbf{C} \mathbf{u}_1 + \lambda_1 \left(1 - \mathbf{u}_1^T \mathbf{u}_1\right)$$

where $\mathbf{C}$ is the covariance matrix of the data set and λ_1 is the strength of the constraint (which we find is the eigenvalue). The first term defines the variance of the data in the direction of $\mathbf{u}_1$, which we would like to maximise so that the transformation retains as much information. The second term is the constraint to keep the vector normalised to 1.

This can also be re-cast as minimising the reconstruction error

$$E = \frac{1}{N} \sum_n \left(\mathbf{x}_n - \tilde{\mathbf{x}}_n\right)^2$$

where $\mathbf{x}_n$ is the input data and $\tilde{\mathbf{x}}_n$ is the reconstruction after applying the transform and it's inverse.

Q 6.6

An auto-encoder has a shape where the first part of the network reduces the input size repeatedly to find a compressed latent space, this is the encoder. From this we mirror these operations to reconstruct the input from this latent space. This can be done with either fully connected or convolutional operations. The optimisation criterion is usually to minimise the reconstruction error (see answer to Q 6.5).

Q 6.7

For dimensionality reduction we train the AE to reconstruct the input but when we want to reduce the dimensionality we utilise only the encoder part of the architecture.

Q 7.3

K-means clustering works better on data that is spherical and with a similar spread among the clusters. When this is not true the clusters found by the algorithm can span multiple real clusters. Good examples are when the clusters are moon shaped, concentric circles, or with wide variance mixed with low variance. See if you can find out more examples of where it fails.

Q 7.5

If we assume that the data has been encoded onto the affinity graph so that W_{ij} represents the similarity of data points x_i and x_j, e.g using a Guassian (RBF) kernel. In this case the node degrees are

$$d_i = \sum_j W_{ij}$$

and the volume of a set A (by a set we mean a cluster of points i labelled together as set A)

$$\mathrm{Vol}(A) = \sum_{i \in A} d_i.$$

The cut of the network into sets A and B will be connections that are removed when it is separated into these sets. This is easy to visualise using a sketch of the network but mathematically is calculated as

$$\mathrm{Cut}(A, B) = \sum_{i \in A} \sum_{j \in B} W_{ij}.$$

Notice we have 2 sums, the first over the data points in set A but the second sum restricts this over data points in set B so it is only connections that span the 2 sets. The normalised cut is the cut normalised by the volumes of the 2 sets

$$\mathrm{Ncut}(A, B) = \frac{\mathrm{Cut}(A, B)}{\mathrm{Vol}(A)} + \frac{\mathrm{Cut}(A, B)}{\mathrm{Vol}(B)}$$

which is the same as

$$\mathrm{Ncut}(A, B) = \mathrm{Cut}(A, B) \left(\frac{\mathrm{Vol}(A) + \mathrm{Vol}(B)}{\mathrm{Vol}(A)\mathrm{Vol}(B)} \right).$$

Q 8.2

The key assumption for the Naive Baye's classifier is that the features are conditionally independent.

Q 8.7

The Evidence Lower Bound (ELBO) arises because the evidence (the denominator in Bayes' rule) is intractable, especially for large dimensional latent spaces. Instead of learning the true posterior $p(z|x)$, which is the conditional probability distribution of the latent space given a input, we instead learn an approximation, the variational distribution. The closer this distribution can get to the true one, the closer our 'lower bound' will be to the true evidence. Whilst we are not directly computing the evidence we have a metric to tell us how close it is based on distributions we do have. Importantly, we can relate the ELBO to the Kullback-Liebler divergence of the true and variational posteriors.