UNIVERSITY of York

BSc, BEng, MEng and MMath Examinations, 2016-2017

Computer Science, Computer Systems and Software Engineering, Computer Science and Mathematics, Computer Science with Artificial Intelligence

Part B

Machine Learning and Applications (MLAP)

Open Examination

Issued at:
15 February 2017

Submission due:
10 May 2017

Your attention is drawn to the Guidelines on Mutual Assistance and Collaboration in the Student's Handbook and the Department's 'Guide to Assessment Policies and Procedures' (http://www.cs.york.ac.uk/exams/statementonassessment/).

All queries on this assessment should be addressed to **James Cussens & Edwin Hancock**.

Your examination number must be written on the front of your submission. You must not identify yourself in any other way.

Should you wish to request an extension see the appropriate section of the Department's 'Guide to Assessment Policies and Procedures'.

- All tasks should be attempted.
- A submission after the deadline will be marked initially as if it had been handed in on time, but the Board of Examiners will normally apply a lateness penalty.
- Your submission should be submitted electronically through the web page http://www.cs.york.ac.uk/submit.
- Your submission must be a **single file**, so zip together all your files and submit the zip archive file.

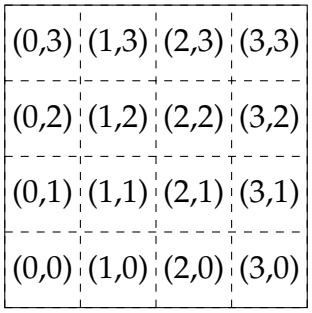

Figure 1: An environment for the agent

- All extra materials for this assessment are available via the VLE. To get them click on 'Assessment' from the VLE MLAP module webpage.
- Be sure to use only your exam number as an identifier in your submission.

1 The EM algorithm

1.1 Problem scenario

For Tasks 1–3 below we consider an agent who moves around between the 16 'cells' depicted in Fig 1. At any point in time, the agent either moves to an adjacent cell or stays where it is. No diagonal moves are allowed. So, for example, if at time point t the agent were in cell $(0,2)$ then at time point $t+1$ it could only be in one of the following cells: $(0,2)$, $(1,2)$, $(0,1)$ or $(0,3)$.

The agent moves probabilistically according to *transition* probabilities. So, for example, there is a transition probability $P(h_{t+1} = (1,2)|h_t = (0,2))$ which, for any time point t, is the probability that the agent arrives in cell $(1,2)$ at time $t+1$ given that it was in cell $(0,2)$ at time t.

In addition to the transition probabilities there is also an *initial* probability for each cell which is the probability that the agent starts in that cell. For example, there is a probability $P(h_1 = (2,3))$.

At each time point, the agent receives a 'reward'. There are three levels of reward: -1, 0 or 1. The reward the agent receives depends on a probability distribution that is specific to the cell the agent is currently visiting. So, for example, there are 3 *emission* probabilities $P(v_t = -1|h_t = (0,2))$, $P(v_t = 0|h_t = (0,2))$ and $P(v_t = 1|h_t = (0,2))$. These three probabilities will sum to one and are independent of t.

If the cells the agent visits are hidden from us then this process is clearly a Hidden Markov Model (HMM). In the following tasks you are required to

estimate the initial, transition and emission probabilities using data. In the following I will use the term *model parameters* to mean the collection of initial, transition and emission probabilities.

1.2 What to submit

For this section (Section 1) you are required to submit a program called `mlap_prog` which can be used to perform all the tasks required. This program can be written in any language but can only use the standard library for that language. For example, my model answer is written in Python and uses the standard Python library modules sys, math and re. It must be possible for me to execute your `mlap_prog` program from the Linux prompt when using a machine in the department's software labs.

The tasks below require you to do some writing. Just put this in your assessment report and make it clear which task it is for.

1.3 How your code will be marked

The mark you get for your code will be determined entirely by (1) its (degree of) correctness and (2) the efficiency of the algorithm. It is the efficiency of the implemented *algorithm* that is important. If one student implemented a given algorithm in C and another that same algorithm in Python then there would be the same mark for efficiency even though the C implementation would be quicker.

1.4 Task 1 [10 marks]

In this task you are required to compute maximum likelihood estimates (MLEs) for model parameters from the data in the file `task1.dat` which is available via the VLE (as indicated above). This data has been generated using initial, transition and emission probabilities which you are not given. This is data for 5 *episodes*. In any particular episode the agent moves around and receives rewards according to the unknown initial, transition and emission probabilities. Each episode ends after a number of time steps decided by me. In the data file, episodes are separated by a blank line. Note that the episodes are of varying lengths.

For each time point you are given the cell that the agent is in and the reward the agent received in that cell. Since you get to see the cells they are not hidden.

Running the following from the Linux command line:

```
mlap_prog 1 task1.dat
```

should print out all the MLE model parameters to standard output (i.e. the terminal). Note that the first argument value of "1" is being used to identify the task number. The format of your output is up to you, just be sure to provide the MLE for each model parameter. If an MLE is not defined be sure to state this.

In your report, explain, in no more than 50 words, how you computed these MLEs.

Distribution of marks for this task Code: 5 marks. Explanation: 5 marks.

1.5 Task 2 [20 marks]

Download the file `task2.dat` from the VLE. The nature of the data in `task2.dat` is different to that in `task1.dat` in that the cell the agent is visiting at any time point is not given to you: it is hidden. Take this data and use the EM algorithm to produce parameter estimates for all model parameters. Use uniform distributions as your starting point for the EM algorithm.

You know that, for example, $P(h_{t+1} = (2,2)|h_t = (0,2)) = 0$ since $(0,2)$ and $(2,2)$ are not adjacent. However, do not adapt the EM algorithm to use this knowledge: just use the standard EM approach for parameter estimation in HMMs. This means you will end up with parameter estimates you know cannot be right. This is not a problem.

Be sure to use forward and backward probabilities in your EM algorithm. Failure to do so will be penalised.

Running the following from the Linux command line:

```
mlap_prog 2 task2.dat
```

should print out all the estimated model parameters to standard output (i.e. the terminal). Note that the first argument value of "2" is being used to identify the task number. You need to also print out the log-likelihood achieved for each iteration of the EM algorithm. Your EM algorithm should terminate if the improvement in log-likelihood is less than 0.01 compared to the previous iteration.

In your report, explain, in no more than 150 words, how you implemented EM.

Distribution of marks for this task Code: 12 marks. Explanation: 8 marks.

1.6 Task 3 [10 marks]

Adapt your EM algorithm so that it starts with randomly chosen model parameters. Do 10 runs with 10 different random starting points.

If you were to enter the following at the Linux command line:

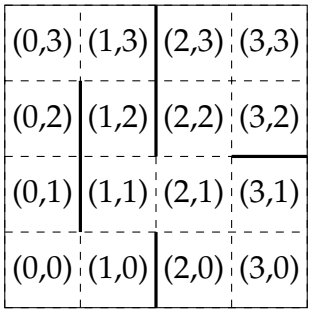

Figure 2: An environment for the agent which has 4 walls

```
mlap_prog 3 task2.dat
```

then the 10 (randomly initialised) EM runs should be printed to standard output. Each particular run should be printed out as in Task 2.

In your report explain why different log-likelihoods are achieved for different EM runs. You should get different behaviour with random initialisation than with the uniform initialisation done in Task 2. Explain this difference in behaviour in your report. Use no more than 200 words for this part of your report.

Distribution of marks for this task Code: 5 marks. Explanation: 5 marks.

1.7 Task 4 [10 marks]

This task is similar to Task 3 except that the environment in which the agent operates has walls between some adjacent cells. This environment is shown in Fig 2. The agent cannot move from cell C_1 to cell C_2 if there is a wall between C_1 and C_2.

You should adapt your EM algorithm to use this information **and** the fact that the agent can only move to adjacent cells. So for example, the estimate your EM algorithm returns for, say, going from $(0, 2)$ to $(1, 2)$ should be zero since there is a wall. Also, for example, the estimate your EM algorithm returns for, say, going from $(0, 2)$ to $(0, 0)$ should also be zero since they are not adjacent.

Running the following from the Linux command line:

```
mlap_prog 4 task2.dat
```

should run this altered EM algorithm. Be sure to print out the log-likelihood at each iteration and the final model parameters.

Explain how your altered your EM algorithm to use the information specified above. Use no more than 20 words to do this.

Distribution of marks for this task Code: 5 marks. Explanation: 5 marks.

2 Manifold learning

2.1 Task 5 [50 marks]

Isomap, the Laplacian Eigenmap, and Locality Preserving Projection (LPP) are three manifold learning techniques which use a neighbourhood graph to capture the structure of the data being studied. They are described in the following papers:

1. Locally Preserving Projection: "Locality Preserving Projection", Xiaofei He and Partha Niyogi, Advances in Neural information Processing Systems 16, pp. 153-160, 2003.

2. Laplacian Eigenmaps: "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation", Mikhail Belkin, Partha Niyogi, Neural Computation, 15, pp. 1373-1396, 2003.

3. Isomap: "A Global Geometric Framework for Nonlinear Dimensionality Reduction", Joshua B. Tenenbaum, Vin de Silva, John C. Langford, Science, 290, pp. 2319-2323, 2000.

Read the papers and answer the following questions on the different methods.

1. Without using mathematics, in qualitative terms describe how each algorithm works. Use diagrams where they help to explain your answer. [10 marks]

2. The methods each utilise a data proximity graph at some stage. Explain the different role of the graph in the three methods. [5 marks]

3. One of the mathematical tools used in these methods is the Laplacian matrix. Describe how this matrix is constructed from a graph, and explain the significance of the Fiedler vector. [5 marks]

4. Suppose you are given a set of data $X = \{\vec{x}_1, ...\vec{x}_N\}$ where $\vec{x}_i$ is the i-th vector which is of length d and there are N such vectors. For each of the methods in turn, give a brief mathematical description of the algorithm, commencing from X. [15 marks]

5. These three methods have been used in the analysis of facial image variations and face recognition. Using Scopus, ISI Web of Science and Google Scholar (either singly or in combination) identify the advantages and disadvantages of the three methods in the face recognition domain. Cite the papers used to source for your answer. [15 marks]

You should be able to supply a complete answer in about 1500 words.