COMP3411/9814 25T1 Assignment 2 Marking Scheme

Assessment for this assignment is mainly subjective marking (Part A does have an automarked component). Subjective marking is your assessment, given these guidelines, based on reading the submitted reports and code files. Please consider providing some feedback for each student submission using the commenting facility in `xmark`.

Part A.

Automarking [3 marks]

This is just FYI — the automarking script will run the submitted network with the trained weights on a held-out test set. Default accuracy is around 86%, and a linear model can get around 92%, so the trained network should be able to do at least as well. The automarking scheme is therefore:

(i) classifier runs on test set and gets classification accuracy $\geq 92\%$ **[1 mark]**

(ii) linearly interpolated classification accuracy from 92% up to maximum (around 97%) **[2 marks]**

Training and validation set performance [3 marks]

For this, and the next section, you will need to refer to the report. Marks will be gained by clearly showing the results and providing reasonable commentary. Marks will be lost by simply describing the results without demonstrating any interpretation (only describing "what happened" without any argument about "why this happened").

(i) a learning curve plot by epochs for classifier training: no mark if plot does not have correctly labelled axes, or if plot does not show expected increase in accuracy over epochs **[1 mark]**

(ii) a correctly plotted and labelled two-class confusion matrix: no mark if this is not correctly shown **[1 mark]**

(iii) the commentary can relate to: the model improves performance over epochs, i.e., it learns, and anything the student highlights in implementation that they believe helped **[0.5 mark]**; anything on the highly skewed class ratio (only approximately 13.9% positive class examples) meaning that absolute accuracy is misleading – e.g., there could be a sentence with some discussion about reducing false negative errors being necessary to improve accuracy, etc. **[0.5 mark]**

Answers to Questions A1 and A2 [3 marks]

(A1) the answer describes and justifies network architecture and parameter selection, mentions experimentation during implementation (possibly using a structured approach such as neural architecture search), and overall displays clear understanding of network configuration choices and possible effects of parameter choice on learning **[2 marks]**

(A2) the student should observe and report that performance without scaling is clearly lower **[0.5 mark]**; in addition they should have checked the input data and observed the difference in value ranges across the variables, which suggests the need for scaling **[0.5 mark]**

Code quality [2 marks]

(i) code is well-structured, e.g., has clear layout with inputs, hidden layer(s), outputs, hyperparameters, train/test setup, result reporting, etc. implemented in a modular, logical and readable way **[1 mark]**

(ii) code is implemented in an "experienced Python programmer" style with consistent use of naming, minimal imports, no 'magic numbers', unnecessary global variables, etc. to reduce the potential for bugs or maintenance issues **[1 mark]**

Part B.

Task 1: Implement Q-learning [4 marks]

(i) the code in "`teach.py`"should implement the function `train_agent(max_total_steps)` (there can be additional parameters). The implementation should adhere to the specification and should be well-written code in the usual sense (see Part A). It should also be clear, either in the code or the report, what was implemented for key choices such as learning rate (α), discount factor (γ), action selection method (e.g., ϵ-greedy, softmax) and so on, as outlined in the specification. **[1.5 marks]**

(ii) the report should include a plot that displays the total reward per episode, which should closely resemble the plot in the specification. **[1 mark]**

(iii) the report should include results for the following:

(a) Average Reward per Episode: expected range is something like from 20 to 60. **[0.5 mark]**

(b) Success Rate: expected range is something like from 80 to 100%. **[0.5 mark]**

(c) Average Steps per Episode: expected range is something like from 20 to 40. **[0.5 mark]**

There will be variation outside these ranges but for anything way off you should check the code and deduct some fraction of a mark for any obvious errors.

Task 2: Agent evaluation [3 marks]

Here the students should report values for the same measures used in Task 1. However, since the agent should have learned a good policy at this point there should be much less variation.

(i) Average Reward per Episode: expect a value close to 60. **[1 mark]**

(ii) Success Rate: expect a value close to 100%. **[1 mark]**

(iii) Average Steps per Episode: expect a value close to 20. **[1 mark]**

There will be some variation in these values but for anything greater than, say, 10% away from the expected value you should check the code and deduct some fraction of a mark for any obvious errors.

Task 3: Q-learning with Teacher Advice [3 marks]

Tasks 3 and 4 are closely related. Marks in Task 3 are allocated mainly for correct implementation in the "`teach.py`" code of the teacher training procedure, and in Task 4 for inclusion in the report of the evaluation results and discussion.

(i) the code in "`teach.py`"should implement the function `train_agent_with_teacher(teacher_q_table, max_total_steps, availability, accuracy)` (there can be additional parameters). Given that the `train_agent()` function was already implemented in Task 1 the first requirement for this task is to implement the function `provide_teacher_advice(teacher_q_table, state, availability, accuracy)` according to the specification. **[1 mark]**

(ii) the `train_agent_with_teacher()` function must also implement the following components:

(a) `teacher_q_table` – is this passed in from a previously trained agent **[0.5 mark]**

(b) `max_total_steps` – value is between 6,000-12,000, not the previous 50,000 **[0.5 mark]**

(c) `availability` – are the values in the specification being used in the iteration **[0.5 mark]**

(d) `accuracy` – are the values in the specification being used in the iteration **[0.5 mark]**

Task 4: Analysis of Teacher Training [4 marks]

This is the most difficult part of the assignment to mark, since you will almost certainly see considerable variation in the heatmaps produced in different submissions. This is probably because the evaluation of learning with teacher training is given much less time than the initial agent learning phase. You will have to use discretion and judgement when assessing student submissions, particularly in the interpretation of the evaluation of teacher training appearing in the second heatmap.

First, this marking guide assumes the code correctly implements the specification (marks for this are allocated in Task 3). If this is not the case, deduct marks for any results or discussion points in Task 4 that are NOT consistent with the implementation as assessed in Task 3.

(i) heatmap plots for training and evaluation are correctly labelled and formatted with accuracy and availability arranged in increasing order along the horiontal and vertical axes, respectively. **[0.5 mark]**

(ii) expect to see relatively little variation across the bottom row of the plot where availability is set to zero, since the teacher is not involved and so accuracy changes should not have any effect. **[0.5 mark]**

(iii) we *may* see in the plots higher average rewards around the "upper-middle" ranges of accuracy and availability (for accuracy, say 0.6-1.0 and for availability, say 0.4-0.8), particularly in the second plot on the evaluation of teacher training. If this is the case it could be interpreted as a good balance of assistance from the teacher plus exploration by the learner. **[1 mark]**

(iv) in the top-right corner, where accuracy and availability are both 1.0, we can expect to see considerable variation between submissions, showing reward that is either relatively high, or relatively low, or neither. If reward is relatively high, it could be interpreted that the high availability of a "well-trained" Q-table enhances training of the learning agent If reward is relatively low, it could be interpreted that the high availability of a "well-trained" Q-table limits the ability of the learning agent to learn from exploration during training. **[1 mark]**

(v) in the top-left corner, where accuracy = 0.2 and availability = 1.0, again there can be considerable variation. If reward is relatively high, it could be interpreted that the high availability of an "adversarial" teacher, which deliberately selects a different action than the best one in the trained Q-table table enhances training of the learning agent by forcing more exploration If reward is relatively low, it could be interpreted that the high availability of an "adversarial" teacher also forces more exploration but the learning agent may not have enough time to learn well, or that it simply degrades training. **[1 mark]**