Module Code
COM00049H

BSc, BEng, MEng and MMath Degrees 2023–24

Open Assessment

Department Computer Science

Module Intelligent Systems: Probabilistic & Deep Learning

Title Open Individual Assessment

Issued Thursday, 21st March 2024 Midday

Submission due Thursday, 23rd May 2024 Midday

Feeedback and Marks due Thursday, 27th June 2024

Instructions:

- All data and any other additional files are available at the PADL VLE site in the **Assessment** section. Your submission should be **a single zip file** named after your exam number, `Yxxxxxxx.zip`, which contains: 1. a single Jupyter notebook `padl.ipynb` (combining code and explanations), 2. a PDF `padl.pdf` of the state of the same notebook after all of its code has been executed (see below for advice on how to export your notebook to PDF), 3. your trained network weights for Question 4, 4. a python script `predict_time.py` that can be used to run your trained network for Question 4. In case of any discrepancies between the contents and output of the Jupyter notebook and the PDF, the former will be used for marking. Make sure you do not use archive formats other than zip. Your code should assume all data files are in the same folder as the notebook from which they are accessed.

- All Python code should be in Python3. Your Jupyter notebook must run correctly when run on Google Colab as practised in the practicals.

- Do not hide code blocks or entire tiles in your notebook, as it slows down the marking process.

- Unless otherwise indicated there are no word limits on your answers, but unnecessarily verbose or irrelevant comments will be marked down.

- Inline comments in the code are no substitute for text tiles with complete

sentences addressing a point.

- To save your colab notebook in PDF format follow these instructions. First, download your notebook (File -> Download -> Download .ipynb) and save it as `padl.ipynb`. Second, create a new blank colab notebook and upload `padl.ipynb` to the session storage in this notebook. Third copy and paste the following commands into a code block in the new empty notebook and execute it:
```
!sudo apt-get install texlive-xetex
texlive-fonts-recommended texlive-plain-generic
!jupyter nbconvert –to pdf /content/padl.ipynb
```
Finally, refresh session storage and then download the PDF file.

All students should submit their answers through the electronic submission system: the Department's Teaching Portal by Thursday, 23rd May 2024 Midday. An assessment submitted after this 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 attention is drawn to the section about Academic Misconduct in your Departmental Handbook.

Any queries you may have on this assessment should be posted on the Discussion Board on the Virtual Learning Environment (VLE) page for Probabilistic & Deep Learning (PADL) in the appropriate discussion area. **No questions will be answered after Thursday, 4th April 2024**.

Note on Academic Integrity

We are treating this online examination as a time-limited open assessment, and you are therefore permitted to refer to written and online materials to aid you in your answers. However, you must ensure that the work you submit is entirely your own, and for the whole time the assessment is live you must not:

- communicate with other students on the topic of this assessment.
- seek advice or contribution from any other third party, including proofreaders, friends, or family members.

We expect, and trust, that all our students will seek to maintain the integrity of the assessment, and of their award, through ensuring that these instructions are strictly followed. Where evidence of academic misconduct is evident this will be addressed in line with the Academic Misconduct Policy and if proven be penalised in line with the appropriate penalty table. Given the nature of these assessments, any collusion identified will normally be treated as cheating/breach of assessment regulations and penalised using the appropriate penalty table (see AM3.3. of the Guide to Assessment).

1 (9 marks) Principal Component Analysis

Download the `PADL-Q1.csv` file from the Assessment section of the PADL VLE site. The file contains 200 rows of numerical data for five variables, $x1$ to $x5$, as listed in the first row of the file.

(a) [3 marks] Apply principal component analysis (PCA) with a number of principal components (PCs) equal to the number of original variables, i.e. $p = 5$, and study the result in order to decide whether you can reduce the dimensionality of the dataset with little to no information loss. Report the minimum number of dimensions D_{min} the new representation will require and briefly explain your choice.

(b) [6 marks] Repeat the PCA analysis using D_{min} number of dimensions (principal components) and show the equations used to compute each principal component from the original variables $x1, \ldots, x5$. List these equations in decreasing order of the variance of the principal component they define.

Marking guidance For each of the parts: one third of the marks for working code, another third for good, informed design choices, and one third for explaining it well.

2 (27 marks) Regression Models

Download the file `PADL-Q2-train.csv` from the Assessment section of the PADL VLE site. The comma-separated file contains data on four variables, x, y, z, w, and *out*. The label for each column is given in the first, header row of the file. The remaining rows contain 80 data points:

x	y	z	w	out
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

Your ultimate goal here is to train and submit a single regression model whose performance will be tested by the exam markers on unseen data not available to you.

Throughout this question you need to use scikit-learn – no marks will be given for the use of PyTorch! Regression should always be evaluated in terms of the R^2 value on out-of-sample data.

(a) [9 marks] You need to demonstrate that you have considered and evaluated a suitable range of basis functions, alongside the possible use of data scaling and/or normalisation.

(b) [9 marks] You need to demonstrate that you have considered, tuned and evaluated a suitable range of linear regression models with respect to the possible use of regularisation and piecewise regression.

(c) [9 marks] You need to implement an appropriate automated procedure that will train all of the above models and select the model expected to perform best on unseen data of the same kind as your training data. Include a code tile at the end of this section of your Jupyter notebook that attempts to test your final choice of model on a data set stored in a file `PADL-Q2-unseen.csv` and compute R^2 for it. The file will have exactly the same format as file `PADL-Q2-train.csv`, including the header, but possibly a different overall number of rows. This means you can use a renamed copy of `PADL-Q2-train.csv` to debug that part of your code, and to produce the corresponding content for your PDF file (in order to demonstrate that this part of the code is in working order).

Breakdown of marks For each of the parts: one third of the marks for working code, another third for good, informed design choices, and one third for explaining it well.

3 (14 marks) Embeddings

Cockney rhyming slang is a social and linguistic phenomenon where, in order to disguise the topic of a conversation from prying ears, one word is replaced with another word or a phrase. One of the possible patterns used is as follows: to disguise a given word X, one finds a pair of semantically related words Y and Z, such that the latter word Z rhymes with the word X. Then in the conversation, the word X is replaced by either the entire phrase 'Y and Z' or by Y alone. For instance, replacing 'stairs' with 'apples and pears' or just 'apples' would give sentences, such as: *I went up the apples*. Similarly, *'He is on the dog and bone' (on the phone)*.

Download the plain text of Sir Arthur Conan Doyle's book 'Adventures of Sherlock Holmes' from this URL: https://www.gutenberg.org/cache/epub/48320/pg48320.txt. Your aim will be to design a fully automated procedure that will make use of this text to ultimately generate candidate rhyming slang phrases of the above mentioned 'A and B' pattern for each word in the following list $L = [$'gold', 'diamond', 'robbery', 'bank', 'police'$]$. To achieve that, take the following steps:

(a) [3 marks] Design and implement a procedure that takes the text of the entire book and extracts all triplets of words where the word in the middle is 'and'.

(b) [3 marks] Design and implement a procedure that takes a word W from the list L and selects all triplets T_i produced in the previous step, such that the last of the three words in the triplet, and the word W share a suffix of length at least 3, e.g. 'old' – 'gold', 'bold' – 'gold', 'snobbery' – 'robbery'.

(c) [8 marks] For each word W and corresponding list of triplets $[T_1, T_2, \ldots]$ selected in the previous step, use word2vec to compute the semantic similarity between the first and third word in the triplet, and sort the triplets in order of decreasing similarity, then return the top 5 triplets. You can use a pretrained word2vec model or you can train it yourself from scratch.

Marking guidance For each of the parts: one third of the marks (rounded up) for working code, another third (rounded up) for good, informed design choices, and one third (rounded down) for explaining it well.

4 (10 marks) Basic MLPs

Consider a multilayer perceptron (MLP) with two inputs and one output, $f : \mathbb{R}^2 \to \mathbb{R}$, that seeks to approximate multiplication of two real numbers, i.e. $f(x, y) \approx x \cdot y$.

(a) [4 marks] Using only linear (fully connected) and ReLU layers, implement an MLP for this task in PyTorch.

(b) [3 marks] Create an appropriate training loop to train the network on random data using a loss function of your choice. You should explicitly comment on what assumptions you are making about the range of the training data.

(c) [3 marks] Evaluate the network in terms of the absolute error in its prediction, i.e. $|f(x, y) - (x \cdot y)|$. You should report a mean error over random samples within the range of the training data but also a generalisation error when tested on inputs outside the range of the training data.

5 (20 marks) Telling the time

Consider the task of telling the time from an image of an analogue clock face. The objective is to estimate the hour, H (an integer in the range $0 \ldots 11$ so $H \in \{0, 1, \ldots, 11\}$) and the minute, M (an integer in the range $0 \ldots 59$ so $M \in \{0, 1, \ldots, 59\}$), from a given image. You are to tackle this as an end-to-end deep learning problem, i.e. train a network that takes an image as input and directly outputs the estimated time.

We have provided you with a training dataset of cartoon clocks (`clocks_dataset.zip` on the VLE). Once unzipped, you will find 10k training images in PNG format, named `0000.png` to `9999.png`. Each image is of size $H = 448$, $W = 448$ and they are RGB colour. Examples from the dataset are shown in Figure 1. The images are procedurally generated with the hour and minute drawn randomly from uniform distributions. Each image is accompanied by a text file containing the corresponding label, named `0000.txt` to `9999.txt`. Each text file contains a single line with the ground truth time in the format `HH:MM`.

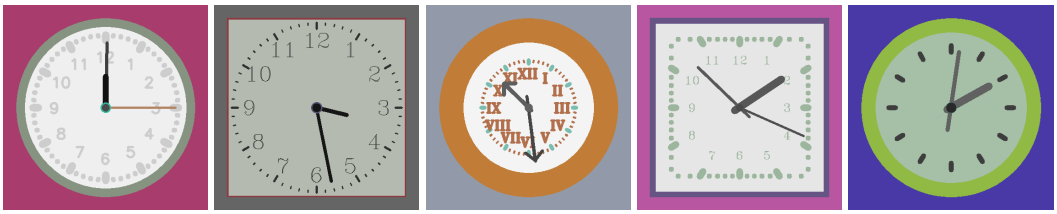

Figure 1: Example images from the training set. Note the wide variation in clock shapes, colours, text and hand appearance. Clocks may or may not have second and/or alarm hands. The ground truth labels associated with each clock are $(H = 0, M = 0)$, $(H = 3, M = 23)$, $(H = 10, M = 24)$, $(H = 1, M = 52)$ and $(H = 2, M = 1)$ respectively.

Your task in this question is to implement and train a network in PyTorch that outputs the time given an image. You may tackle this problem as either classification or regression. Your network will be evaluated on an unseen test. The performance metric will be the absolute difference in minutes between the estimated and ground truth time. For example, if the ground truth is 11:59 and you predict 00:01 then the error will be 2 minutes. The test images will be generated by exactly the same procedure as the training images and the overall performance will be measured by the median of the absolute differences over the whole test set.

Your goal is for this median error to be as close to zero as possible. You do not have access to the test set, but you need to include a python script `predict_time.py` that I can import which contains a function `predict(images)` as explained in the assessment section of the VLE. This function should load your pretrained network from the weights you supply, pass the input images through the network and return `times`. The input `images` is a $B \times 3 \times 448 \times 448$ PyTorch tensor containing a batch of B images, each with an RGB image of size 448×448 - i.e. the same format as the training data. Intensity values will be in the range $(0, 1)$. Any preprocessing or normalisation that you need to apply to the images must be inside

the `predict` function. The output `times` is a $B \times 2$ tensor, where `times[:,0]` contains the estimated hours for each image (an integer between 0 and 11) and `times[:,1]` contains the estimated minutes for each image (an integer between 0 and 59).

Your notebook must include your training and validation code along with discussion and justification for all design decisions. You are not allowed to use transfer learning on a pretrained network (i.e. you must train your network from scratch) and your saved network weights must not exceed 20MiB.

(a) [5 marks] Create a dataloader for the clocks dataset. You are free to preprocess or augment the images in any way you deem suitable but should explain your decisions in comments or text blocks.

(b) [4 marks] Design and implement an appropriate network architecture.

(c) [3 marks] Choose and justify an appropriate loss function.

(d) [3 marks] Plot training and validation losses and use this to justify hyperparameter choices.

(e) [5 marks] Up to 5 marks are available for performance on the unseen test set. A median error $<$5 minutes will score 5 marks. Partial marks will be awarded for larger errors.

6 (20 marks) Generative models

The final task uses the same dataset as Question 5. However, this time you will only use the images, not the labels. **The goal is to train a generative image model** such as a Generative Adversarial Network, Variational Autoencoder or Diffusion Model to create realistic clock images. Again, you may preprocess the images however you like and can augment to provide more images for training if you wish.

You should include a code tile that generates 8 random samples from your generative model. i.e. randomly sample from your latent space 8 times, pass these through your generator network and display the resulting images. Ensure that your PDF file properly displays these images.

You should also include a code tile that performs latent space interpolation between two samples. Generate two random samples from the latent space as before, then linearly interpolate 5 intermediate latent vectors and display all 7 resulting images in order (i.e. the first randomly sampled clock image, then the 5 interpolated samples, then finally the second randomly sampled clock image). You would expect that the middle image looks something like an average between the start and end images, while all should be plausible. Ensure that your PDF file contains these images and choose an example where the two random samples are visually different so that the effect of the interpolation is clear.

(a) [5 marks] Appropriate network architecture chosen with justification.

(b) [5 marks] Appropriate preprocessing applied, training loop implementation and choice of training hyperparameters.

(c) [5 marks] Successfully generate 8 random clock images (marks awarded depending on quality of synthesised images).

(d) [5 marks] Successfully interpolate between two random clock images in latent space.

END OF PAPER