CM4107 AI Coursework 1

Nirmalie Wiratunga Ike Nkisi-Orji Anjana Wijekoon

Outline

The **aim** of this coursework is to further investigate the Artificial Neural Network (ANN) and the k-Nearest Neighbour (kNN) Python implementation from labs and lectures. Specifically you will carry out a comparative study to understand the behaviour of hyper parameters, and the impact of adding new data for both training and testing.

Completing this coursework successfully will require that you develop an understanding of the principal theories, concepts and methods used in the development of machine learning algorithms. Further you will gain experience of evaluating and interpreting results from two popular machine learning algorithms applied to an image recognition dataset (in this case the MNIST handwritten characters dataset).

A secondary aim is to test your ability to apply and transfer your knowledge to create a new model by combining the ANN and kNN implementations.

Submission deadline: 25th Oct 4pm

Weight Allocation: This coursework comprises of 5 sub tasks and carries 50% of the weight towards the final grade. Weights assigned to each task is an indication of its individual contribution to the final grade.

Feedback: Students will be given initial verbal feedback in the form of an outline / expected solution one week after submission of Part 1 (1st Nov). A provisional grade will be made available 20 days after the coursework submission date.

1 Comparative Study with the MNIST Dataset

The MNIST dataset introduced in the ANN lecture and lab and can be downloaded from the moodle CW area. For more information on the dataset refer to http://yann.lecun.com/exdb/mnist/

1.1 Task 1: Artificial Neural Network (ANN)

Provide one paragraph to explain your understanding of each hyper-parameter's role on a neural network's performance. You should consider the following hyper-parameters:

- epoch (the number of training iterations on the full dataset);
- batch size (the number of examples considered within a single epoch before updating network parameters); and
- learning rate (the amount of moderation used to manage by how much we adjust the weights).

Carry out an evaluation to study each of these hyper-parameters on the basic neural network (implemented in the ANN lab), find the optimal hyper-parameter settings for:

- Number of epochs
- Learning rate
- Batch size

A graph should be provided for each hyper-parameter, comparing at least five different values for each. The y-axis of the graph should be accuracy, while the x-axis is the parameter setting.

A second paragraph should then be provided to discuss the results, and to explain why you think the parameter is optimal in these conditions. You should consider reporting your results using sentences such as: "we can see that with increasing epochs the overall accuracy of the model decreased because <your reason>; whilst increasing learning rate we observe that <your observation>" etc.

Task (weight 1): Maximum one page with graphs must be organised as follows:

- Paragraph detailing students understanding of hyperparamaters.
- A graph for each hyperparameter comparing the different values.
- Paragraph explaining the results.

1.2 Task 2: kNN

Provide one paragraph detailing your understanding of the kNN algorithm, the role of the hyper-parameter k and the differences between unweighted and weighted voting in kNN.

Carry out an evaluation to compare weighted and unweighted kNN algorithms (implemented in lab 4) at different values of k. Provide a single graph, to present your results from the comparison. You should aim to include at least

five different values of k. The y-axis will detail accuracy, while the x-axis will be different values of k.

A second paragraph should be provided, discussing any additional insights you have about the results.

Task (weight 1): Maximum one page which needs to be organised as follows:

- Paragraph detailing understanding of weighted and unweighted kNN.
- Graph comparing weighted and unweighted kNN at different values of k.
- Paragraph explaining the results.

1.3 Task 3: Creating New Data to Augment Training Data

The MNIST dataset is a rich set of examples of how people write numbers. There are many variations to how a number might be written and an ANN must learn all these nuances. Of course the more variations it can learn from the more robust it will become at deployment.

In this task you will explore how to create further variations on the given set of training examples. A popular idea is to take the existing training data, and create further examples from these by rotating them clockwise and anticlockwise, by some small degree. Thereafter these are added to the training data to be used by the machine learning algorithm (here ANN or kNN) . Your task is to explore the impact on accuracy as you add more data by rotating existing training data.

You are provided with a function **rotate_image** (download this from the coursework area). Use this function to add new data to the training set and provide a comparative study on the impact of adding new data to the existing training data.

Task (weight 1): Maximum one page with graphs must be organised as follows:

- Paragraph to explain your strategy for using the rotated images in your experiment to augment your training data. You could refer to any code modifications you may have made to integrate the new training data. Please do not use this space to repeat the functions you were already provided on Moodle.
- A graph to illustrate the impact of the augmented training with increasing rotations on algorithm accuracy for ANN and kNN.
- Paragraph to discuss your findings. Here you should explore whether or not the algorithms benefited from the extra data and also explain why / why not.

1.4 Task 4: Testing on your handwriting

In this task you will explore the accuracy of the two algorithms on a small sample of your own handwritten numbers. You can create images using any image editing or painting software you like (e.g. GIMP is a free open source alternative for Windows, Mac and Linux). You can even use a pen on paper and photograph your writing with a smartphone. The only requirement is that the image **MUST** be saved as 28x28 PNG file. You can save several such files in a sub folder and make sure to name them such that the character number appears just before the .PNG (e.g. pen_image_3.png for an image of the letter '3')

Download the function **get_my_images** on Moodle and use it to convert your PNGs into an array input for the ANN.

Task (weight 1): Maximum one page which needs to be organised as follows:

- Provide a paragraph to specify how you created your images and how you integrated them in to test data. If you tried different mediums for image creation then state these (e.g. using different pen tips, etc.). Also state any modifications you made to integrate the images into the test data. Please do not use this space to repeat the functions you were provided on Moodle.
- Explain your observations from testing on your own handwriting data.

1.5 Task 5: Combining ANN and kNN

Using the optimal parameters from previous parts of the coursework (Task 1 & Task 2); develop a hybrid system which improves data representation. The idea is that you make use of the hidden layer activation for each training instance and use that as input into the kNN.

This should be introduced with a paragraph discussing how you enabled kNN to operate on the ANN hidden layer representation. You may also want to include a figure to help your explanation.

A table should be provided, comparing accuracy of the hybrid system against the kNN and neural network. This will demonstrate that the improvement in accuracy (if any) of the hybrid system is due to improved representation (from the hidden layer) as gained from the network.

Task (weight 1): Maximum one page which needs to be organised as follows:

- Paragraph detailing how you went about combining the ANN with kNN. It will be useful to provide a figure to explain your strategy.
- Table comparing the results of the ANN, kNN and hybrid.
- Short description to interpret your results.

2 How to Submit

Please follow these guidelines:

- Submit by the due date.
- Use the **WRITEUP** dropbox to submit your write-up using the Word template file (max 5 pages). Please also use Turnitin to evidence that you submission contains non plagarised material.
- Use the **ZIP** dropbox to submit a zip of your any ipynb files that evidence your coding effort. Files should be named appropriately to link with the different sections of this coursework e.g. task1.ipynb, task2.ipynb, task3.ipynb, etc.
- All files should be submitted as a single zip file named using your surname-firstname-matriculation_number (e.g. Turing_Alan_00991876).
- Please also use Turnitin to evidence that your submission contains non-plagarised material.

3 How grades are aggregated from CW1 & CW2

Figure 1: Aggregation of grades from CW1 and CW2.

CW1 (50%) \ CW2 (50%)	A	B	C	D	E	F	NS
A	A	B	B	C	D	D	NS
B	B	B	C	C	D	D	NS
C	B	C	C	C	D	D	NS
D	C	C	C	D	D	E	NS
E	D	D	D	D	E	E	NS
F	D	D	D	E	E	F	NS
NS	NS	NS	NS	NS	NS	NS	NS