DEPARTMENT OF
COMPUTING SCIENCE
UNIVERSITY OF ALBERTA

CMPUT 366: Assignment #4

Due at 10pm on December 5, 2019

Abstract

For this assignment use the following consultation model:

- you can discuss assignment questions and exchange ideas with other CMPUT 366 students;
- you must list all members of the discussion in your solution;
- you may **not** share/exchange/discuss written material and/or code;
- you must write up your solutions individually;
- you must fully understand and be able to explain your solution in any amount of detail as requested by the instructor and/or the TAs.

Anything that you use in your work and that is not your own creation must be properly cited by listing the original source. Failing to cite others' work is plagiarism and will be dealt with as an academic offence.

First name: ________________________________

Last name: ________________________________

CCID: ___________________________`@ualberta.ca`

Collaborators: ______________________________

Your mark: _____ **out of 81**

Submission

The assignment you downloaded from eClass is a single ZIP archive which includes this document as a PDF file as well as its LaTeX source and support files for the programming questions. Your answers are to be submitted electronically via eClass. Your submission must be a single ZIP file containing a single PDF file with all of your answers as well as your Python code for the coding questions. To generate the PDF file you can do any of the following:

1. insert your answers into the provided LaTeX source file between `\begin{answer}` and `\end{answer}`. Then run the source through LaTeX to produce a PDF file;

2. write your answers in the blank spaces under each question. Make sure you write as legibly as possible and scan the written pages properly for we cannot give you any points if we cannot read your hand-writing;

3. use your favourite text processor and type up your answers there. Make sure you number your answers in the same way as the questions are numbered in this assignment.

1 Supervised Learning: Short Answers

1.1

[2 points] Why is error on the training dataset not a good estimate for generalization performance?

1.2

[2 points] Give an expression for the value of a single rectified linear unit (relu), with inputs x_1, x_2, weights w_1, w_2, and bias b.

1.3

[2 points] Why are convolutional networks more efficient to train than fully-connected feed-forward networks?

1.4

[2 points] What is overfitting?

1.5

[2 points] List two strategies to avoid overfitting.

1.6

[3 points] What is the output of a *convolution* operation on the matrix $\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ using kernel $\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$?

1.7

[3 points] What is the output of a 2×2 *maximum pooling* operation on the matrix $\mathbf{X} = \begin{bmatrix} 8 & 1 & 2 \\ 3 & 4 & 1 \\ 2 & 3 & 9 \end{bmatrix}$?

2 Supervised Learning: Errors & Classifiers

Consider the following dataset about the edibility of an imaginary family of plants:

example	spots	rings	leaves	edible
e_1	0	3	6	1
e_2	0	6	3	0
e_3	0	7	2	0
e_4	0	9	3	0
e_5	0	10	5	1
e_6	1	0	2	0
e_7	1	1	4	0
e_8	1	2	2	1
e_9	1	2	3	1
e_{10}	1	4	2	1

2.1

[3 points] What is the 0/1 error on the above dataset of the following hypothesis?

$$\widehat{edible}(e) = \begin{cases} 1 & \text{if } spots(e) = 1, \\ 0 & \text{otherwise.} \end{cases}$$

2.2

[3 points] What is the mean squared error on the above dataset of the following hypothesis?

$$\widehat{edible}(e) = \begin{cases} 0.8 & \text{if } rings(e) \leq 4, \\ 0.2 & \text{otherwise.} \end{cases}$$

2.3

[5 points] What is the log loss of a decision tree which has only a single test *spots* $= 0$? and whose two leaf nodes make the best probabilistic predictions on the target value *edible*? Use the binary logarithm and show your work.

3 Deep Learning

The MNIST dataset is a set of images of handwritten digits labelled by their actual digit. We will operate on two versions of this dataset:

1. `upper-left:` with the images shifted 2 pixels to the upper left;
2. `bottom-right:` with the images shifted 2 pixels to the bottom right.

These questions require the use of TensorFlow. You may need to install Tensorflow using the following command: `pip3 install tensorflow`

3.1

[10 points] Implement a fully-connected feed-forward neural network for classifying MNIST images according to the digit that they represent by editing the `mlp2` function in the provided `cnn.py` file.

The network should have two hidden layers: one with 128 rectified linear ('relu') units, and one with 64 rectified linear units. The output should be a fully-connected layer of 10 units with the softmax activation. The `cnn.py` file contains an example implementation of a network with a single hidden layer in the `mlp1` function that you may template from. The `main` function will test your program for you; you may add any tests that you like. It will also create a file called `examples.png` that contains example images from the two test sets.

The function should train the network using the training features `train_x` and labels `train_y`, and then evaluate the accuracy of the trained network on two different test sets: `test1_x`, `test1_y`, and `test2_x`, `test2_y`. This is demonstrated in `mlp1`.

We will run your code by importing the `cnn` module and calling `mlp2`, so it is important that your code follow these naming conventions.

Submit all of your code including provided boilerplate files in a single zip file.

3.2

[30 points] Implement a convolutional neural network for classifying MNIST images by editing the `cnn` function in the provided `cnn.py` file.

The network should have the following architecture:

- a layer of 32 convolutional units with a kernel size of 5×5 and a stride of $1, 1$;
- a max-pooling layer with a pool size of 2×2 and a stride of $2, 2$;
- a layer of 64 convolutional units with a kernel size of 5×5 and the default stride;
- a max-pooling layer with a pool size of 2×2 and the default stride;
- a `Flatten` layer (to reshape the image from a 2D matrix into a single long vector);
- a layer of 512 fully-connected relu units;
- a layer of 10 fully-connected softmax units (the output layer).

Submit all of your code including provided boilerplate files in a single zip file.

3.3

[2 points] What was the accuracy of your trained 2-hidden-layer feedforward network on the two test sets?

3.4

[2 points] What was the accuracy of your trained convolutional neural network on the two test sets?

3.5

[10 points] Did one of your implemented networks perform substantially better on one of the test sets than the other network did? If so, why? If not, why not?