COMP 2211 Final Exam - Spring 2024 - HKUST

Date:	May 22, 2024 (Wednesday)
Time Allowed:	3 hours, 8:30–11:30 am
Instructions:	

1. This is a closed-book, closed-notes examination.
2. There are **9** questions on **12** pages (including this cover page, honor code, and 4 blank pages at the end).
3. Write your answers in the space provided in black/blue ink. *NO pencil please, otherwise you are not allowed to appeal for any grading disagreements.*
4. All programming codes in your answers must be written in the Python version as taught in the class.
5. For programming questions, unless otherwise stated, you are **NOT** allowed to define additional classes, helper functions and use global variables, nor any library functions not mentioned in the questions.

Student Name	SOLUTIONS & MARKING SCHEME
Student ID	
Venue and Seat Number	

Problem 1 [10 points] True/False Questions

Indicate whether the following statements are true or false by putting **T** or **F** in the given table. You get 1 point for each correct answer.

Question	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)
Answer	T	F	F	T	F	T	F	F	F	T

Scheme:

- 1 point for giving each correct answer. 10 points in total.

Problem 2 [10 points] Advanced Python: Image Processing with NumPy

Solution:

(a)
```python
def apply_circle_mask(img, center_x, center_y):
    indices = np.arange(512)
    x_dist = indices - center_x
    y_dist = indices - center_y
    dist = np.sqrt(x_dist ** 2 + y_dist[:, None] ** 2)
    img_masked = img * (dist <= 100)
    return img_masked
```

(b)
```python
def img_flatten_conv_1d(img, v):
    zeros = np.zeros((512, 1))
    img_padded = np.concatenate((zeros, img, zeros), axis=1)
    img_flat = img_padded.reshape(-1)
    img_flat_conv = np.convolve(img_flat, v, 'valid')
    img_flat_conv = np.concatenate(([0], img_flat_conv, [0]))
    img_conv = img_flat_conv.reshape(512, 514)[:, 1:-1]
    return img_conv

v = [1/3, 1/3, 1/3]
img_blur = img_flatten_conv_1d(img, v)
img_blur = img_flatten_conv_1d(img_blur.T, v).T
```

Scheme:

(a) If no explicit loop is used, 1 point for each of the following:

- correct broadcasting to create 2D distance array;
- correct distance values;
- correct mask (0.5 point for the opposite mask);
- correct final result.

If explicit loops are used, 1 point in total if the final result is correct.

(b) If no explicit loop is used, 1 point for each of the following:

- correct padding before flattening the image array (0.5 point if padding is applied after flattening);
- correct flattening of the image array;
- correct dimensions of inputs (1D arrays) to `np.convolve`;
- correct values of inputs to `np.convolve`;
- correct padding after `np.convolve`;
- correct reshaping back to 2D image and removing padding (0.5 each).

If explicit loops are used, 1 point in total if the final result is correct.

Problem 3 [12 points] Naïve Bayes, K-Nearest Neighbors and Perceptron

Solution:

(a) (i)

$$\frac{1}{\sqrt{2\pi(0.5)^2}}\exp\left(-\frac{(36-36.5)^2}{2(0.5^2)}\right) = 0.4839$$

(ii)

$$\frac{1}{\sqrt{2\pi(15)^2}}\exp\left(-\frac{(85-90)^2}{2(15^2)}\right) = 0.0252$$

(iii)

$$(0.4839)(0.0252)(0.5) = 0.0061$$

(iv)

$$\frac{0.0061}{0.0061+0.0000036} = 0.9994$$

(b) (i) This will predict 5/6 correctly if uniform distance (prediction for Patient 5 is wrong). Also, if inverse distance is used, the prediction for Patient 3 will also be wrong.

(ii) It is possible that the data will be folded in such a way that the class opposite to the true label is typically the majority class e.g. (Patient 1, Patient 2), (Patient 3, Patient 4), (Patient 5, Patient 6) . In this case, most folds will have 50% accuracy.

The performance may be even worse as there is one pair of samples from opposite classes with minimum pairwise Euclidean distance (Patient 3 and Patient 5).

(iii) This is not acceptable because it is possible that the training set for the KNN model will have the class opposite to the true label as its majority class. In this case, the prediction for all samples will be wrong.

(c) Yes, because in that case the data is linearly separable.

Scheme:

(a) (i) 1.5 points for giving the correct answer.

(ii) 1.5 points for giving the correct answer.

(iii) 1.5 points for giving the correct answer.

(iv) 1.5 points for giving the correct answer.

(b) (i) 1.5 point for giving the correct reason.

(ii) 1.5 point for giving the correct reason.

(iii) 1.5 point for giving the correct reason.

(c) 0.5 point for stating "Yes", i.e., perceptron model will make a good prediction for the sample. 1 point for giving the correct explanation.

Problem 4 [11 points] Multi-layer Perceptron

Solution:

(a) (i) (I) For any hidden layer and the output layer, the parameters include the weight and the bias.

$$(n+1) \times l_1 + \sum_{k=1}^{L-1} l_{k+1} \times (l_k + 1) + m \times (l_L + 1)$$

(II) Model A: 67; Model B: 53.

(ii) Model A is better because it has more parameters and therefore more expressive.

(b) To add non-linearity to the neural network model so that it has more powerful modeling capability.

(c) The problem is that we cannot learn the parameters using gradient descent since the gradients are 0 almost everywhere. We can solve the problem while approximating a hard threshold by scaling up the weights in a sigmoid activation function. For example, $\sigma(cx)$ is steeper than $\sigma(x)$ and more similar with the binary step function, for $c > 1$.

(d) (i) Hidden layer $O_{j1} = 0.2$, $O_{j2} = 0$
Output layer $O_{k1} = 0.48$, $O_{k2} = 0.52$

(ii) $w_5' = w_5 - \delta_{k1}\eta O_{j1} = 0.1 - (-1298)(0.4)(0.2) = 0.1104$
$w_7' = w_7 - \delta_{k2}\eta O_{j1} = 0 - (1298)(0.4)(0.2) = -0.0104$
$\delta_{j1} = O_{j1}(1-O_{j1})(\delta_{k1}w_5+\delta_{k2}w_7) = 0.2(1-0.2)(-0.1298(0.1)+0.1298(0)) = -0.0021$
$w_1' = w_1 - \delta_{j1}\eta x_1 = 0.2 - (0.4)(-0.0021)(1) = 0.2008$

Scheme:

(a) (i) (I) 1.5 points for giving the correct formula.

(II) 1 point for each correct answer. 2 points in total.

(ii) 0.5 point for stating Model A is better. 1 point for giving the correct explanation. 1.5 points in total.

(b) 1 point for giving the correct answer for why we use activation functions in multi-layer perceptron.

(c) 1 point for stating the problem. 1 point for explaining how to make use of the sigmoid function to avoid the problem. 2 points in total.

(d) (i) 0.5 point for giving each correct output. 1.5 points in total.

(ii) 0.5 point for giving each correct weight value. 1.5 points in total.

Problem 5 [13 points] Digital Image Processing

Solution:

(a) (b)-(f): image (horizontal) flipping because the histogram is the same as the original image.
(c)-(d): binary thresholding because there are only two values 0 and 255 in the histogram
(a)-(e): contrast stretching because the intensity value in the histogram has been stretched to a wider range.

(b) After the first iteration μ_1=21 , μ_2=128, T = 74.5 The resulting images are

0	0	0
0	0	0
255	255	255

Compared to regular image thresholding algorithms, Otsu's method has advantages (i) can automatically determine the threshold value T (ii) the resulting threshold value is reproducible. Given the same image, two researchers using Otsu's algorithm must arrive at the same threshold.

(c) Resulting image:

32	16	16	32	64	64	32
4	2	2	4	8	8	4
4	2	2	4	8	8	4
32	16	16	32	64	64	32
128	128	128	128	128	128	128
128	128	128	128	128	128	128
32	16	16	32	64	64	32

(d) (i) smoothing, (ii) vertical edge detection, (iii) sharpening.

(e) No, because the image flipping is a global operation on the image, but convolution with a 3×3 kernel is a local operation. Concretely, the 3×3 kernels can only capture the input value of 3×3 neighbors, but the flipping requires the pixel value information at a longer distance. The longest dependency distance can be 64.

Scheme:

(a) 1 for stating each transformation correctly. 3 points in total. 0.5 point for giving each correct pairing. 1.5 points in total.

(b) 1 point for giving the correct resulting threshold. 1 point for giving the correct result image. 1 point for stating the advantage of using Otsu's method. 3 points in total.

(c) 0.05 for giving each correct value (40 values). 2 points in total.

(d) 0.5 point for stating each effect of convolving with the given kernel correctly. 1.5 points in total.

(e) 0.5 point for stating it is impossible to design a 3×3 kernel and apply it to flip the 64×64 image. 1.5 points for giving the explanation.

Problem 6 [13 points] Dilated Convolution and Dropout

Solution:

(a) Dilated_Convolution

TODO #	Answer
1	`((kernel.shape[0] - 1) * dilation_rate) // 2` 1.5 points
2	`(input_array.shape[0] - kernel.shape[0] * dilation_rate + 2 * pad) // stride + 1` 1.5 points
3	`(input_array.shape[1] - kernel.shape[1] * dilation_rate + 2 * pad) // stride + 1` 1.5 points
4	`np.arange(0, output_rows * stride, stride) + i * dilation_rate` 1.5 points
5	`np.arange(0, output_cols * stride, stride) + j * dilation_rate` 1.5 points
6	`kernel[i, j] * padded_input[np.ix_(input_row_indices, input_col_indices)]` 2.5 points

(b) Dropout

```
(np.random.rand(input_array.shape[0],input_array.shape[1]) < p) / p  # 3 points
```

Problem 7 [18 points] Convolutional Neural Network

Solution:

(a) (i) 2

(ii) `16, 16, 64`

`8, 8, 64` `36928`

`1024`

(iii) The model has overfit the training dataset. Overfitting occurs when a model is too complex and learns the noise in the training data rather than the underlying patterns.

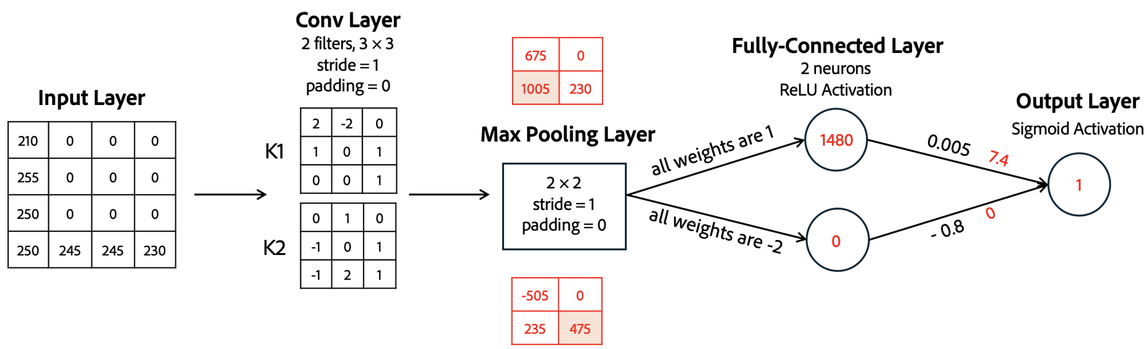
Dropout can be used to mitigate the problem. Dropout is a technique where, during training, some neurons in the network are randomly dropped out (i.e., their outputs are set to zero) with a certain probability, typically 0.2 or 0.5. This means that, at each training iteration, a different subset of neurons is randomly selected to be "dropped out."

Dropout helps prevent overfitting in several ways **(full marks to any of the following)**:

- Reducing capacity: By randomly dropping out neurons, the network's capacity is reduced, making it less prone to overfitting. With fewer neurons, the network has fewer opportunities to memorize the training data.
- Forcing feature sharing: Dropout encourages feature sharing among neurons. When a neuron is dropped out, the network must rely on other neurons to make predictions, which promotes feature sharing and reduces overfitting.
- Preventing complex co-adaptations: Dropout breaks the complex co-adaptations between neurons, which can lead to overfitting. By randomly dropping out neurons, the network is forced to learn simpler, more generalizable representations.
- Improving generalization: Dropout can be seen as a form of data augmentation. By randomly dropping out neurons, the network is forced to generalize to new, unseen situations, which improves its ability to generalize to new data.
- Reducing the risk of over-reliance on a single neuron: Dropout prevents the network from relying too heavily on a single neuron or a small group of neurons. This reduces the risk of overfitting, as the network is forced to use multiple neurons to make predictions.
- Ensemble-like behavior: Dropout can be seen as an ensemble method, where multiple sub-networks are trained simultaneously. Each sub-network is a different subset of neurons, and the final prediction is an ensemble of these sub-networks. This ensemble-like behavior improves generalization and reduces overfitting.

The answer is not unique.

(b) Answer:



(c) (i) For any dimension (let's denote it generically as D): Output_D = floor((Input_D + 2 × Padding - Kernel_D) / Stride) + 1. Therefore, we have:
Output_keyframe = floor((32 + 2 * 0 - 3) / 1) + 1 = 30
Output_width = floor((400 + 2 * 2 - 3) / 2) + 1 = 201
Output_height = floor((300 + 2 * 2 - 3) / 2) + 1 = 151
The output shape is (30, 201, 151, 100).

(ii) There are 100 kernels in the shape of (3, 3, 3). Therefore, the number of weight parameters is 1000 × 3 × 3 × 3 x 3 = 8100.

(iii) There is 1 bias parameter per kernel. So the total biases is 100.

Scheme:

(a) (i) 1 point for giving the correct padding size.

(ii) 1 point for giving each correct shape (3 shape values). 1 point for giving the correct number of parameters. 4 points in total.

(iii) 1 point for stating the model has overfit the training dataset. 1 point for explaining what does the problem usually occur. 1 point for describing a way to mitigate the problem. 3 points in total.

(b) 1 point for giving each correct feature map (2 feature maps). 1 point for giving each feature in the fully-connected layer (2 features). 1 point for giving the correct output. 5 points in total.

(c) (i) 2 points for giving the correct output shape after this layer.

(ii) 1.5 points for giving the correct number of weight parameters.

(iii) 1.5 points for giving the correct total number of biases.

Problem 8 [10 points] Minimax and Alpha-Beta Pruning

Solution:

(a) (i) Answer:

Nodes	Score
A	2
B	2
C	4
D	5

(ii) Answer:

Edge	Alpha	Beta
CH	2	2
DJ	3	2

(iii) Record depth information to distinguish paths.

(b) (i) Answer:

A	(2,4)
B	(0,3)
C	(-1,3)
D	(1,1)
E	(0,-2)

(ii) No. The values that the first and second player are trying to maximize are independent. Therefore, the principle for pruning in alpha-beta pruning—that a worse outcome for one player implies a better outcome for the other—no longer applies. For instance, in the case where $U_A(s) = U_B(s)$ for all nodes, the problem reduces to searching for the max-valued leaf, which could appear anywhere in the tree.

Scheme:

(a) (i) 0.25 point for giving each correct numeric value. 1 point in total.

(ii) 0.5 point for giving each correct edge/numeric value. 3 points in total.

(iii) 1 point for giving a way to find the shortest path to victory.

(b) (i) 0.5 point for giving each pair of values. 2.5 points in total.

(ii) 0.5 point for stating "No". 1 point for explaining why and 1 point for giving an example. 2.5 points in total.

Problem 9 [3 points] Ethics of Artificial Intelligence

Solution:

- Data Ethics
- Fair AI model (or avoiding AI model bias)
- AI model monitoring and maintenance.

Scheme:

- 1 point for giving each area correctly. 3 points in total.

-------------------- END OF PAPER --------------------