SC4001 Exam 2023-2024 Sem 1

1. (a) Give brief answers to the following. Each part carries 2 marks.

(i) What is the total number of learnable parameters in a 2-layer network with 5 neurons in each layer and receiving 2-dimensional inputs.

Answer

Given

- layer = 2
- neuron = 5
- input = 2
- bias = 1

layer 1

- neuron * (input + bias) = 5 * (2 + 1) = 5 * 3 = 15 parameters

layer 2

- neuron * (neuron + bias) = 5 * (5 + 1) = 5 * 6 = 30 parameters

Total = 15 + 30 = 45 parameters

(ii) Write a tensor with the shape [2, 1, 3] and integer elements, as in Python.

Answer

[[[1., 2., 3.]], [[7., 8., 9.]]]

(iii) One wants to initialize all the weights in a 3-layer deep neural networks to 0.4. Is this a good idea? And why?

Answer

- No, it is not a good idea.

- If all weights are the same, every neuron in a layer will compute the same gradient during backpropagation.

(iv) You want to solve a classification problem with a training sample of 50 patterns. But the training loss was high. Then you decide to train with 1000 samples. Is this good approach? And why?

Answer

- Yes, it is a good approach.

- A larger training set gives the model more examples to learn from, improving its ability to generalize to unseen data.

1. (a) cont

(v) You started training a neural network but the loss was neither decreasing or increasing. State two possible reasons.

Answer

- Learning rate is too low or too high.
- The gradients become too small or too large.

(vi) State two ways to handle local minima problem in gradient descent learning.

Answer

- Use SGD or mini-batch GD
- Using adaptive learning rate methods (e.g., Adam)

(vii) State one advantage and one disadvantage of using a small batch size for training.

Answer

- Helps the local minima problem
- Slower training, slower convergence

(viii) State how you can fix underfitting in your neural network.

Answer

- Train longer (more epochs)
- Decrease learning rate or batch size

(16 marks)

1. (b) The output $\boldsymbol{y} \in \boldsymbol{R}^K$ of a linear neuron is given by $\boldsymbol{y} = \boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b}$ where $\boldsymbol{x} \in \boldsymbol{R}^n$ is input, and $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight matrix and bias vector of the layer, respectively. Given a target vector $\boldsymbol{d} \in \boldsymbol{R}^K$, the square error loss J of the layer is given $J = (\boldsymbol{d} - \boldsymbol{y})^T(\boldsymbol{d} - \boldsymbol{y})$.

(i) Write the expressions for $\nabla_{\boldsymbol{y}}J$, $\nabla_{\boldsymbol{W}}J$ and $\nabla_{\boldsymbol{b}}J$.

(5 marks)

Answer

$$J = (d-y)^T(d-y)$$
$$= \|d-y\|^2$$
$$\nabla_{\boldsymbol{y}}J = \frac{\partial J}{\partial y}$$
$$= -2(d-y)$$

$$\nabla_{\boldsymbol{W}}J = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial w}$$
$$y = W^Tx + b$$
$$\frac{\partial y}{\partial w} = x$$
$$\nabla_{\boldsymbol{W}}J = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial w}$$
$$= -2(d-y)x^T$$

$$\nabla_{\boldsymbol{b}}J = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial b}$$
$$y = W^Tx + b$$
$$\frac{\partial y}{\partial b} = 1$$
$$\nabla_{\boldsymbol{b}}J = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial b}$$
$$= -2(d-y)$$

Summary

$$\nabla_{\boldsymbol{y}}J = -2(d-y)$$
$$\nabla_{\boldsymbol{W}}J = -2(d-y)x^T$$
$$\nabla_{\boldsymbol{b}}J = -2(d-y)$$

(ii) If L_2-norm weight regularization penalty $\beta\|\boldsymbol{W}\|^2$ is added to the loss J, write the expressions for $\nabla_{\boldsymbol{W}}J_1$ and $\nabla_{\boldsymbol{b}}J_1$ is the regularized loss. β is the penalty parameter.

(4 marks)

Answer

$$J_1 = (d-y)^T(d-y) + \beta\|W\|^2$$
$$\nabla_W(\beta\|W\|^2) = 2\beta W$$
$$\nabla_{\boldsymbol{W}}J_1 = -2(d-y)x^T + 2\beta W$$

Regularization $\beta\|W\|^2$ does not depend on bias, gradients unchanged

$$\nabla_{\boldsymbol{b}}J_1 = -2(d-y)$$

Summary

$$\nabla_{\boldsymbol{W}}J_1 = -2(d-y)x^T + 2\beta W$$
$$\nabla_{\boldsymbol{b}}J_1 = -2(d-y)$$

2. The 3-layer feedforward neural network shown in Figure Q2 receives two-dimensional inputs $(x_1, x_2) \in \boldsymbol{R}^2$ and produces one-dimensional output y. The first hidden layer consists of three neurons and the second hidden layer consists of two neurons. All hidden neurons have $Tanh$ activations functions and the output neuron is a logistic regression neuron. The weights of the networks are initialized as indicated in the figure and all the biases are initialized to 0.5 (not shown).

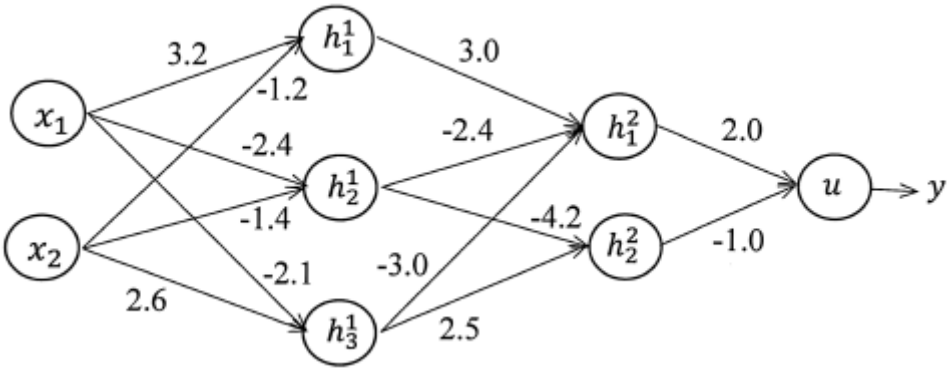

Figure Q2

The network is trained to produce a desired output $d = 0$ for an input $x = \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix}$. You are to perform one iteration of stochastic gradient descent learning with the example $(\boldsymbol{x}, d)$. Give your answers rounded up to two decimal places.

2. (a) Write initial weight matrices $\boldsymbol{W}$ and bias vectors $\boldsymbol{b}$, connected to the three layers. (3 marks)

Answer

$$W_1 = \begin{pmatrix} 3.2 & -2.4 & -2.1 \\ -1.2 & -1.4 & 2.6 \end{pmatrix} \qquad b_1 = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

$$W_2 = \begin{pmatrix} 3.0 & 0.0 \\ -2.4 & -4.2 \\ -3.0 & 2.5 \end{pmatrix} \qquad b_2 = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$W_3 = \begin{pmatrix} 2.0 \\ -1.0 \end{pmatrix} \qquad b_3 = 0.5$$

2. (b) Find the synaptic inputs $\boldsymbol{u}$ and outputs $\boldsymbol{h}$ of the two hidden layers.

(4 marks)

Answer

$$W_1 = \begin{pmatrix} 3.2 & -2.4 & -2.1 \\ -1.2 & -1.4 & 2.6 \end{pmatrix} \qquad b_1 = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

$$W_2 = \begin{pmatrix} 3.0 & 0.0 \\ -2.4 & -4.2 \\ -3.0 & 2.5 \end{pmatrix} \qquad b_2 = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

Input $x = \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix}$

Synaptic input to h_1,

$$u_1 = \boldsymbol{W}_1{}^T x + b_1 = \begin{pmatrix} 3.2 & -1.2 \\ -2.4 & -1.4 \\ -2.1 & 2.6 \end{pmatrix} \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} (3.2)(1) + (-1.2)(2) \\ (-2.4)(1) + (-1.4)(2) \\ (-2.1)(1) + (2.6)(2) \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} 0.8 \\ -5.2 \\ 3.1 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 1.3 \\ -4.7 \\ 3.6 \end{pmatrix}$$

Output of h_1,

$$h_1 = tanh(u_1) = \frac{e^u - e^{-u}}{e^u + e^{-u}} = \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix}$$

Synaptic input to h_2,

$$u_2 = \boldsymbol{W}_2{}^T h_1 + b_2 = \begin{pmatrix} 3.0 & -2.4 & -3.0 \\ 0.0 & -4.2 & 2.5 \end{pmatrix} \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} (3)(0.86) + (-2.4)(-1) + (-3.0)(1) \\ (0)(0.86) + (-4.2)(-1) + (2.5)(1) \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} 1.98 \\ 6.7 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 2.48 \\ 7.20 \end{pmatrix}$$

Output of h_2,

$$h_2 = tanh(u_2) = \frac{e^u - e^{-u}}{e^u + e^{-u}} = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix}$$

<u>Summary</u>

synaptic input $u_1 = \begin{pmatrix} 1.3 \\ -4.7 \\ 3.6 \end{pmatrix}$

output $h_1 = \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix}$

synaptic input $u_2 = \begin{pmatrix} 2.48 \\ 7.20 \end{pmatrix}$

output $h_2 = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix}$

2. (c) Find the output $\boldsymbol{y}$ and the cross-entropy at the output layer.

(4 marks)

Answer

$W_3 = \begin{pmatrix} 2.0 \\ -1.0 \end{pmatrix}$ $\quad b_3 = 0.5$

Desired output $\quad d = 0$

$h_2 = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix}$

Synaptic input to y,

$$u_3 = \boldsymbol{W}_3{}^T h_2 + b_3 = (2 \quad -1)\begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix} + (0.5)$$
$$= \big((2)(0.99) + (-1)(1)\big) + (0.5)$$
$$= 0.98 + 0.5 = 1.48$$

Output of y,

$$y = sigmoid(u_3) = \frac{1}{1 + e^{-u}} = \frac{1}{1 + e^{-1.48}} = 0.81$$

cross-entropy,

$$\mathrm{J} = -d\log(y) - (1-d)\log(1-y)$$
$$= -0\log(0.81) - (1-0)\log(1-0.81)$$
$$= -\log(1-0.81)$$
$$= -\log(0.19) = 1.66$$

Summary

output: $\quad y = 0.81$

cross-entropy: $\quad \mathrm{J} = 1.66$

2. (d) Find the derivatives $f'(\boldsymbol{u})$ at the two hidden layers with respect to synaptic input $\boldsymbol{u}$, where f is the $Tanh$ activation function.

(2 marks)

Answer

Derivatives of $Tanh$

$$y = f(u) = tanh(u) = \frac{e^{+u} - e^{-u}}{e^{+u} + e^{-u}}$$

$$f'(u) = 1 - y^2$$

Hidden layer h_1,

$$h_1 = tanh(u_1) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}} = \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix}$$

$$f'(u_1) = 1 - {h_1}^2 = \begin{pmatrix} 1 - (0.86)^2 \\ 1 - (1.00)^2 \\ 1 - (-1.00)^2 \end{pmatrix} = \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix}$$

Hidden layer h_2,

$$h_2 = tanh(u_2) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}} = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix}$$

$$f'(u_2) = 1 - {h_2}^2 = \begin{pmatrix} 1 - (0.99)^2 \\ 1 - (1.00)^2 \end{pmatrix} = \begin{pmatrix} 0.02 \\ 0 \end{pmatrix}$$

Summary

derivatives $f'(u_1) = \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix}$

$f'(u_2) = \begin{pmatrix} 0.02 \\ 0 \end{pmatrix}$

2. (e) Find gradients $\nabla_{\boldsymbol{u}} J$ of the cost $\boldsymbol{u}$ with respect to activations $\boldsymbol{u}$, at the three layers. (6 marks)

Answer

Backpropagation for FFN:

Desired output $d = 0$

$y = sigmoid(u_3) = 0.81$

Outer layer u_3,

$$\nabla_{\boldsymbol{u_3}} J = -(d - y) = -(0 - 0.81) = 0.81$$

Second hidden layer u_2,

$$W_3 = \begin{pmatrix} 2.0 \\ -1.0 \end{pmatrix} \qquad f'(u_2) = \begin{pmatrix} 0.02 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{u_2}} J = W_3(\nabla_{\boldsymbol{u_3}} J) \cdot f'(u_2) = \begin{pmatrix} 2.0 \\ -1.0 \end{pmatrix}(0.81) \cdot \begin{pmatrix} 0.02 \\ 0 \end{pmatrix} = \begin{pmatrix} (2)(0.81) \\ (-1)(0.81) \end{pmatrix} \cdot \begin{pmatrix} 0.02 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1.62 \\ -0.81 \end{pmatrix} \cdot \begin{pmatrix} 0.02 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.03 \\ 0 \end{pmatrix}$$

First hidden layer u_1,

$$W_2 = \begin{pmatrix} 3.0 & 0.0 \\ -2.4 & -4.2 \\ -3.0 & 2.5 \end{pmatrix} \qquad f'(u_1) = \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{u_1}} J = W_2(\nabla_{\boldsymbol{u_2}} J) \cdot f'(u_1) = \begin{pmatrix} 3.0 & 0.0 \\ -2.4 & -4.2 \\ -3.0 & 2.5 \end{pmatrix} \begin{pmatrix} 0.03 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} (3.0)(0.03) + (0.0)(0) \\ (-2.4)(0.03) + (-4.2)(0) \\ (-3.0)(0.03) + (2.5)(0) \end{pmatrix} \cdot \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0.09 \\ -0.07 \\ -0.09 \end{pmatrix} \cdot \begin{pmatrix} 0.26 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.02 \\ 0 \\ 0 \end{pmatrix}$$

Summary

$$\nabla_{\boldsymbol{u_3}} J = 0.81$$

$$\nabla_{\boldsymbol{u_2}} J = \begin{pmatrix} 0.03 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{u_1}} J = \begin{pmatrix} 0.02 \\ 0 \\ 0 \end{pmatrix}$$

2. (f) Find gradients $\nabla_{\boldsymbol{W}} J$, and $\nabla_{\boldsymbol{b}} J$ of the cost J with respect to weights $\boldsymbol{W}$, and biases $\boldsymbol{b}$, respectively, at three layers.

(6 marks)

Answer

Outer layer u_3,

$$h_2 = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix} \qquad \nabla_{\boldsymbol{u_3}} J = 0.81$$

$$\nabla_{\boldsymbol{W_3}} J = h_2 (\nabla_{u_3} J)^T = \begin{pmatrix} 0.99 \\ 1.00 \end{pmatrix} 0.81 = \begin{pmatrix} (0.99)(0.81) \\ (1.00)(0.81) \end{pmatrix} = \begin{pmatrix} 0.80 \\ 0.81 \end{pmatrix}$$

$$\nabla_{\boldsymbol{b_3}} J = \nabla_{\boldsymbol{u_3}} J = 0.81$$

Second hidden layer u_2,

$$h_1 = \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix} \qquad \nabla_{\boldsymbol{u_2}} J = \begin{pmatrix} 0.03 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{W_2}} J = h_1 (\nabla_{u_2} J)^T = \begin{pmatrix} 0.86 \\ -1.00 \\ 1.00 \end{pmatrix} (0.03 \quad 0) = \begin{pmatrix} (\ 0.86)(0.03) & (\ 0.86)(0) \\ (-1.00)(0.03) & (-1.00)(0) \\ (\ 1.00)(0.03) & (\ 1.00)(0) \end{pmatrix}$$

$$= \begin{pmatrix} 0.03 & 0 \\ -0.03 & 0 \\ 0.03 & 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{b_2}} J = \nabla_{\boldsymbol{u_2}} J = \begin{pmatrix} 0.03 \\ 0 \end{pmatrix}$$

First hidden layer u_1,

$$x = \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix} \qquad \nabla_{\boldsymbol{u_1}} J = \begin{pmatrix} 0.02 \\ 0 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{W_1}} J = x (\nabla_{u_1} J)^T = \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix} (0.02 \quad 0 \quad 0) = \begin{pmatrix} (1)(0.02) & (1)(0) & (1)(0) \\ (2)(0.02) & (2)(0) & (2)(0) \end{pmatrix}$$

$$= \begin{pmatrix} 0.02 & 0 & 0 \\ 0.04 & 0 & 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{b_1}} J = \nabla_{\boldsymbol{u_1}} J = \begin{pmatrix} 0.02 \\ 0 \\ 0 \end{pmatrix}$$

Summary

$$\nabla_{\boldsymbol{W_3}} J = \begin{pmatrix} 0.80 \\ 0.81 \end{pmatrix} \qquad \nabla_{\boldsymbol{b_3}} J = 0.81$$

$$\nabla_{\boldsymbol{W_2}} J = \begin{pmatrix} 0.03 & 0 \\ -0.03 & 0 \\ 0.03 & 0 \end{pmatrix} \qquad \nabla_{\boldsymbol{b_2}} J = \begin{pmatrix} 0.03 \\ 0 \end{pmatrix}$$

$$\nabla_{\boldsymbol{W_1}} J = \begin{pmatrix} 0.02 & 0 & 0 \\ 0.04 & 0 & 0 \end{pmatrix} \qquad \nabla_{\boldsymbol{b_1}} J = \begin{pmatrix} 0.02 \\ 0 \\ 0 \end{pmatrix}$$

3. Figure Q3 depicts a network that consists of two convolutional layers and a fully connected layer. The size of input or output volume is represented as $D \times H \times W$, where D is the number of channels, and $H \times W$ is the spatial size.

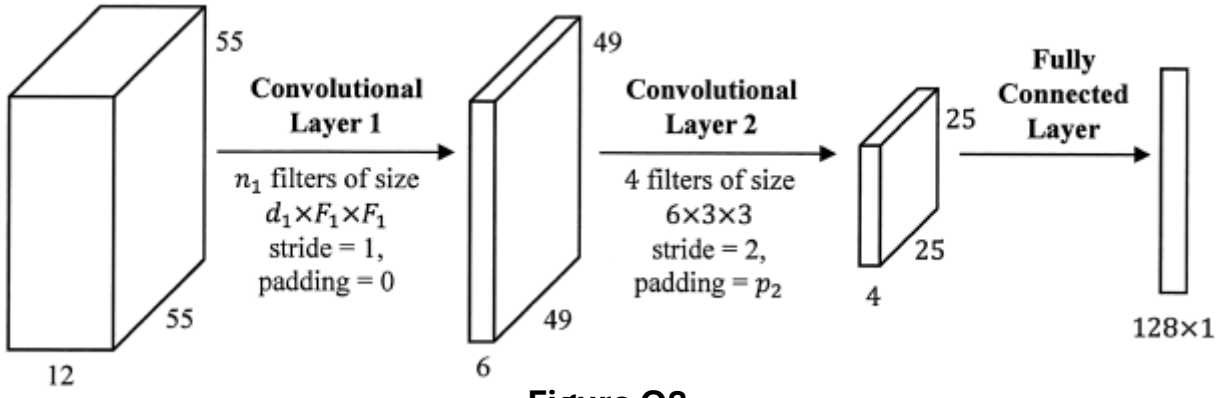

Figure Q3

3. (a) (i) Give the values of n_1, d_1, F_1 and p_2.

(4 marks)

Answer

Convolution Layer 1

- Since output size: 6x49x49, $n_1 = 6$
- Since input size: 12x55x55, $d_1 = 12$

$$\text{Output} = \frac{N - F + 2P}{S} + 1$$

$$49 = \frac{55 - F_1 + 2(0)}{1} + 1$$

$$48 = 55 - F_1$$

$$F_1 = 7$$

- Output = 49
- N = 55
- S = 1
- P = 0

Convolution Layer 2

$$\text{Output} = \frac{N - F + 2P}{S} + 1$$

$$25 = \frac{49 - 3 + 2(p_2)}{2} + 1$$

$$48 = 46 + 2(p_2)$$

$$p_2 = 1$$

- Output = 25
- N = 49
- S = 2
- F = 3

Summary

- $n_1 = 6$
- $d_1 = 12$
- $F_1 = 7$
- $p_2 = 1$

3. (a) (ii) Calculate the total number of parameters in each layer, namely the Convolutional Layer 1, Convolutional Layer 2 and Fully Connected Layer. Be reminded to account for the bias terms.

(4 marks)

Answer

Convolution Layer 1

- filter size = $d_1 \times F_1 \times F_1$
- $d_1 = 12$
- $F_1 = 7$

- convolution filters = 6
- $n_1 = 6$

Parameters per filer = $(d_1 * F_1 * F_1) + 1$ = (12 * 7 * 7) + 1 = 589
Total parameters = param * n_1 = 589 * 6 = 3534

Convolution Layer 2

- $n_2 = 4$
- $d_2 = 6$
- $F_2 = 3$

Parameters per filer = $(d_2 * F_2 * F_2) + 1$ = (6 * 3 * 3) + 1 = 55
Total parameters = param * n_2 = 55 * 4 = 220

Fully Connected Layer

- input flattened = 4 * 25 * 25 = 2500
- output = 128
- bias = 128

Total parameters = (input * output) + bias = (2500 * 128) + 128 = 320128

Total Parameters Summary

Convolution Layer 1 = 3534
Convolution Layer 2 = 220
Fully Connected Layer = 320128

3. (b) (i) Assume the same input volume $12 \times 55 \times 55$ and output volume $6 \times 49 \times 49$, replace the Convolution Layer 1 with a depthwise separable convolution, i.e., "depthwise + pointwise" convolutions. Assume stride = 1 and padding = 0. State the number of filters and the size of filter for each of the depthwise and pointwise convolution layers.

(4 marks)

Answer

Depthwise convolution layer

- Since input size: 12x55x55, number of filters = 12
- Since filter size: 6x7x7, size of filter = 1x7x7

Pointwise convolution layer

- Since output size: 6x49x49, number of filters = 6
- Since input size: 12x55x55, size of filters = 12x1x1

Summary

Depthwise convolution layer

- number of filters = 12
- size of filter = 1x7x7

Pointwise convolution layer

- number of filters = 6
- size of filters = 12x1x1

3. (b) (ii) Calculate the FLOPs of the original Convolution Layer 1, and the FLOPs of the new depthwise separable convolution designed by you in Q3(b)(i). Finally, compute the ratio of the FLOPs between the new and original layers.

(4 marks)

Answer

From Q3(b)(i),

Depthwise convolution layer

- number of filters $D_1 = 12$
- size of filter $F = 7$

Pointwise convolution layer

- Since output size: $6 \times 49 \times 49$, $D_2 * H_2 * W_2 = 6 * 49 * 49$

Original Convolution Layer 1

$$\text{FLOPs}_{\text{standard}} = (2 * D_1 * F^2) * D_2 * H_2 * W_2$$

$= 2 * 12 * 7^2 * 6 * 49 * 49 = 16{,}941{,}456$

Depthwise separable convolution

$$\text{FLOPs}_{\text{depthwise}} = (2 * F^2) * D_1 * H_2 * W_2 = (2 * 7^2) * 12 * 49 * 49 = 2{,}823{,}576$$

$$\text{FLOPs}_{\text{pointwise}} = (2 * D_1) * D_2 * H_2 * W_2 = (2 * 12) * 6 * 49 * 49 = 345{,}744$$

$$\text{FLOPs}_{\text{separable}} = \text{FLOPs}_{\text{depthwise}} + \text{FLOPs}_{\text{pointwise}}$$

$= 2{,}823{,}576 + 345{,}744 = 3{,}169{,}320$

Ratio of the FLOPs

$$\text{Ratio} = \frac{1}{D_2} + \frac{1}{F^2} = \frac{1}{6} + \frac{1}{7^2} = \frac{55}{294} = 0.187$$

Summary

FLOPs of the original Convolution Layer 1 = 16,941,456

FLOPs of the new depthwise separable convolution = 3,169,320

Ratio of the FLOPs (new/original) layers = 0.187

3. (b) (iii) You want to apply batch normalization in your network. Explain why you should not choose a very small mini-batch size during your training.

(3 marks)

Answer

- Unstable estimates of mean and variance
- Inconsistent behaviour across batches
- Poor model generalization

3. (c) The statements below are all related to autoencoders. Answer "TRUE" or "FALSE" to the following statements. Each part carries 1 mark.

Answer

F (i) The primary goal of an autoencoder is to achieve high classification accuracy on labeled data.

T (ii) Autoencoders can be used as a pre-training step for deep networks in situations where labeled data is limited.

F (iii) Stacked autoencoders are created by training multiple autoencoders in parallel, with each one independently processing the input data.

T (iv) Overcomplete autoencoders have a latent space dimension that is larger than the input dimension.

F (v) Sparse autoencoders utilize a loss function that encourages the network to have many activations close to one in the bottleneck layer.

F (vi) Autoencoders with a symmetrical architecture have different layers and neuron counts in the encoder and decoder parts, up to the bottleneck.

(6 marks)

4. (a) Consider a Jordan-type recurrent neural network (RNN) that receives 2-dimensional input patterns $\boldsymbol{x} \in \boldsymbol{R}^2$ and has one hidden layer. The RNN has two neurons in the hidden layer (which are initialized to zeros) and one neuron in the output layer. The hidden layer neurons have *Tanh* activation functions and the output layer neurons use *Sigmoid* activation functions.

The weight matrices $\boldsymbol{U}$ connecting the input to the hidden layer, the top-down recurrence weight matrix $\boldsymbol{W}$, and $\boldsymbol{V}$ connecting the hidden output to the output layer are given by

$$\boldsymbol{U} = \begin{pmatrix} 0.2 & 0.3 \\ 0.8 & 0.9 \end{pmatrix}, \boldsymbol{W} = \begin{pmatrix} 2.0 & 1.0 \end{pmatrix} \text{ and } \boldsymbol{V} = \begin{pmatrix} 0.2 \\ -0.2 \end{pmatrix}$$

All bias connections to neurons are set to 0.2. The output layer is initialized to an output of 1.0.

Find the output of the network for an input sequence $\big(\boldsymbol{x}(1), \boldsymbol{x}(2)\big)$. Provide your answers rounded to four decimal places.

$$\boldsymbol{x}(1) = \begin{pmatrix} 1.5 \\ 1.0 \end{pmatrix} \text{ and } \boldsymbol{x}(2) = \begin{pmatrix} -3.0 \\ 2.0 \end{pmatrix}$$

(6 marks)

Answer

$$\boldsymbol{U} = \begin{pmatrix} 0.2 & 0.3 \\ 0.8 & 0.9 \end{pmatrix}, \boldsymbol{W} = \begin{pmatrix} 2.0 & 1.0 \end{pmatrix} \text{ and } \boldsymbol{V} = \begin{pmatrix} 0.2 \\ -0.2 \end{pmatrix}$$

$$b = \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}, \text{ and } c = b = 0.2$$

$$y = 1.0$$

$$\boldsymbol{x}(1) = \begin{pmatrix} 1.5 \\ 1.0 \end{pmatrix} \text{ and } \boldsymbol{x}(2) = \begin{pmatrix} -3.0 \\ 2.0 \end{pmatrix}$$

$$\phi(u) = \tanh(u) \quad = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

$$\sigma(u) = \text{sigmoid}(u) = \frac{1}{1 + e^{-u}}$$

4. (a) cont

Answer

$$\phi(u) = \tanh(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

$$\sigma(u) = \text{sigmoid}(u) = \frac{1}{1 + e^{-u}}$$

At $t = 1, \boldsymbol{x}(1) = \begin{pmatrix} 1.5 \\ 1.0 \end{pmatrix}$,

$\boldsymbol{h}(t) = \phi(\boldsymbol{U}^T \boldsymbol{x}(t) + \boldsymbol{W}^T \boldsymbol{y}(t-1) + \boldsymbol{b})$

$\boldsymbol{h}(1) = tanh(\boldsymbol{U}^T \boldsymbol{x}(1) + \boldsymbol{W}^T \boldsymbol{y}(0) + \boldsymbol{b})$

$$= \tanh\left(\begin{pmatrix} 0.2 & 0.8 \\ 0.3 & 0.9 \end{pmatrix}\begin{pmatrix} 1.5 \\ 1.0 \end{pmatrix} + \begin{pmatrix} 2.0 \\ 1.0 \end{pmatrix} 1.0 + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right)$$

$$= \tanh\left(\begin{pmatrix} (0.2)(1.5) + (0.8)(1) \\ (0.3)(1.5) + (0.9)(1) \end{pmatrix} + \begin{pmatrix} (2)(1) \\ (1)(1) \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right)$$

$$= \tanh\left(\begin{pmatrix} 1.10 \\ 1.35 \end{pmatrix} + \begin{pmatrix} 2 \\ 1 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \tanh\begin{pmatrix} 3.30 \\ 2.55 \end{pmatrix} = \begin{pmatrix} 0.9973 \\ 0.9879 \end{pmatrix}$$

$\boldsymbol{y}(t) = \sigma(\boldsymbol{V}^T \boldsymbol{h}(t) + c)$

$\boldsymbol{y}(1) = sigmoid(\boldsymbol{V}^T \boldsymbol{h}(1) + c)$

$$= \text{sigmoid}\left((0.2 \quad -0.2)\begin{pmatrix} 0.9973 \\ 0.9879 \end{pmatrix} + 0.2\right)$$

$$= \text{sigmoid}((0.2)(0.9973) + (-0.2)(0.9879) + 0.2) = \text{sigmoid}(0.2019) = 0.5503$$

At $t = 2, \boldsymbol{x}(2) = \begin{pmatrix} -3.0 \\ 2.0 \end{pmatrix}$,

$\boldsymbol{h}(t) = \phi(\boldsymbol{U}^T \boldsymbol{x}(t) + \boldsymbol{W}^T \boldsymbol{y}(t-1) + \boldsymbol{b})$

$\boldsymbol{h}(2) = tanh(\boldsymbol{U}^T \boldsymbol{x}(2) + \boldsymbol{W}^T \boldsymbol{y}(1) + \boldsymbol{b})$

$$= \tanh\left(\begin{pmatrix} 0.2 & 0.8 \\ 0.3 & 0.9 \end{pmatrix}\begin{pmatrix} -3.0 \\ 2.0 \end{pmatrix} + \begin{pmatrix} 2.0 \\ 1.0 \end{pmatrix} 0.5503 + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right)$$

$$= \tanh\left(\begin{pmatrix} (0.2)(-3) + (0.8)(2) \\ (0.3)(-3) + (0.9)(2) \end{pmatrix} + \begin{pmatrix} (2)(0.5503) \\ (1)(0.5503) \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right)$$

$$= \tanh\left(\begin{pmatrix} 1.0 \\ 0.9 \end{pmatrix} + \begin{pmatrix} 1.1006 \\ 0.5503 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}\right) = \tanh\begin{pmatrix} 2.3006 \\ 1.6503 \end{pmatrix} = \begin{pmatrix} 0.9801 \\ 0.9289 \end{pmatrix}$$

$\boldsymbol{y}(t) = \sigma(\boldsymbol{V}^T \boldsymbol{h}(t) + c)$

$\boldsymbol{y}(2) = sigmoid(\boldsymbol{V}^T \boldsymbol{h}(2) + c)$

$$= \text{sigmoid}\left((0.2 \quad -0.2)\begin{pmatrix} 0.9801 \\ 0.9289 \end{pmatrix} + 0.2\right)$$

$$= \text{sigmoid}((0.2)(0.9801) + (-0.2)(0.9289) + 0.2) = \text{sigmoid}(0.2102) = 0.5524$$

Summary

- Output of the network:
  - y(1) = 0.5503
  - y(2) = 0.5524

4. (b) Select the correct option (A, B, C or D) for each question.

A. Both statements are TRUE.
B. Statement I is TRUE, but statement II is FALSE.
C. Statement I is FALSE, but statement II is TRUE.
D. Both statements are FALSE.

Answer

Answer

B (i) Statement I: The self-attention mechanism in Transformers allows each token in the input sequence to focus on different parts of the sequence when producing the output.

T

Statement II: In Transformers, the number of self-attention heads is always fixed at one for all models and applications.

F

(2 marks)

C (ii) Statement I: The Transformer model uses convolutional layers to capture local patterns within the sequence data.

F

Statement II: Layer normalization is a critical component in the Transformer's architecture, helping stabilize the activations throughout the network.

T

(2 marks)

A (iii) Statement I: The "query", "key", and "value" matrices in the self-attention mechanism of Transformers are all derived from the same initial input embeddings.

T

Statement II: In the multi-head self-attention mechanism, different heads can potentially learn to attend to different parts or aspects of the input sequence.

T

(2 marks)

4. (c) The Transformer architecture employs a mechanism known as "positional encoding" to account for the order of words or tokens in a sequence.

(i) Explain how sinusoidal positional encoding is computed and justify why it might be beneficial over other potential positional encoding methods.

(5 marks)

Answer

- *Sinusoidal positional encoding*
  - *interweaves the two signals (*sine for even indices and cosine for odd indices*)*

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

where pos is the position and i is the dimension,

- Beneficial
  - *Each position is uniquely encoded and the encoding can deal with sequences longer than any sequence seen in the training time.*

4. (ii) Based on the sinusoidal positional encoding, calculate the positional encoding values for dimension [0, 10] when the position of a word in the sequence is 5 (assuming the initial position is 0) and the dimension of the embeddings is 512. Provide your answers rounded to four decimal places. (4 marks)

Answer

Given $pos = 5$ and $d_{model} = 512$

At dimension 0, $2i = 0$ thus $i = 0$, therefore $PE_{(pos,2i)} = PE_{(5,0)} = \sin\left(\frac{5}{10000^{\frac{0}{512}}}\right)$= -0.9598

At dimension 1, $2i + 1 = 1$ thus $i = 0$, therefore $PE_{(pos,2i+1)} = PE_{(5,1)} = \cos\left(\frac{5}{10000^{\frac{0}{512}}}\right)$= 0.2837

At dimension 2, $2i = 2$ thus $i = 1$, therefore $PE_{(pos,2i)} = PE_{(5,2)} = \sin\left(\frac{5}{10000^{\frac{2}{512}}}\right)$= -0.9939

At dimension 3, $2i + 1 = 3$ thus $i = 1$, therefore $PE_{(pos,2i+1)} = PE_{(1,3)} = \cos\left(\frac{5}{10000^{\frac{2}{512}}}\right)$= 0.1107

At dimension 4, $2i = 4$ thus $i = 2$, therefore $PE_{(pos,2i)} = PE_{(1,4)} = \sin\left(\frac{5}{10000^{\frac{4}{512}}}\right)$= -0.9982

At dimension 5, $2i + 1 = 5$ thus $i = 2$, therefore $PE_{(pos,2i+1)} = PE_{(1,5)} = \cos\left(\frac{5}{10000^{\frac{4}{512}}}\right)$= -0.0595

At dimension 6, $2i = 6$ thus $i = 3$, therefore $PE_{(pos,2i)} = PE_{(1,6)} = \sin\left(\frac{5}{10000^{\frac{6}{512}}}\right)$ = -0.9750

At dimension 7, $2i + 1 = 7$ thus $i = 3$, therefore $PE_{(pos,2i+1)} = PE_{(1,7)} = \cos\left(\frac{5}{10000^{\frac{6}{512}}}\right)$= -0.2221

At dimension 8, $2i = 8$ thus $i = 4$, therefore $PE_{(pos,2i)} = PE_{(1,8)} = \sin\left(\frac{5}{10000^{\frac{8}{512}}}\right)$= -0.9277

At dimension 9, $2i + 1 = 9$ thus $i = 4$, therefore $PE_{(pos,2i+1)} = PE_{(1,9)} = \cos\left(\frac{5}{10000^{\frac{8}{512}}}\right)$= -0.3733

At dimension 10, $2i = 10$ thus $i = 5$, therefore $PE_{(pos,2i)} = PE_{(1,10)} = \sin\left(\frac{5}{10000^{\frac{10}{512}}}\right)$= -0.8600

4. (d) Explain the concept of "mode collapse" in the context of GANs and discuss its implications on the quality of the generated data.

(4 marks)

Answer

- Mode collapse is a common training problem in GANs where the generator learns to produce limited varieties of outputs, even when the input (noise vector) varies.

- Instead of capturing the full diversity of the training data, the generator repeatedly outputs identical or nearly identical samples, effectively "collapsing" to a few modes.

- Implications
  - Generated samples lack variation, even when input noise is different.

  - The generator becomes overly specialized to a few patterns, reducing its ability to generalize to unseen data or edge cases.

  - Mode collapse often correlates with oscillations or divergence in GAN training, where the generator and discriminator fail to reach equilibrium.