Jawahar Sai Nathani
NetId: 735000830

CSCE 636: Deep Learning
Assignment 2
(Answers to the non-programming part Q1 - Q5)

1

Ans. Given, $E_{in}(w)$, need to calculate $\nabla E_{in}(w)$

$$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N}(tanh(w^T x_n) - y_n)^2$$

$$\begin{aligned}
\nabla E_{in}(w) &= \frac{d}{dw}E_{in}(w) \\
&= \frac{1}{N}\sum_{n=1}^{N}\frac{d}{dw}(tanh(w^T x_n) - y_n)^2 \\
&= \frac{1}{N}\sum_{n=1}^{N}2.(tanh(w^T x_n) - y_n)\frac{d}{dw}(tanh(w^T x_n)) \\
&= \frac{1}{N}\sum_{n=1}^{N}2.(tanh(w^T x_n) - y_n)(1 - tanh^2(w^T x_n))\frac{d}{dw}(w^T x_n) \\
&= \frac{2}{N}\sum_{n=1}^{N}(tanh(w^T x_n) - y_n)(1 - tanh^2(w^T x_n)).x_n
\end{aligned}$$

Hence Proved

- When $w \to \infty$, **tanh** function tends to 1.
- As $w \to \infty$, $1 - tanh^2(w^T x_n) \to \infty$. Therefore, gradient $\nabla E_{in}(w)$ also becomes zero. When the gradient goes to zero, there will be no change in weights and it would be difficult to optimize the perception, resulting in poor training.

2

Ans. Given weight and input matrices:

$$W^1 = \begin{bmatrix} 0.1 & 0.2 \\ 0.3 & 0.4 \end{bmatrix} \qquad W^2 = \begin{bmatrix} 0.2 \\ 1 \\ -3 \end{bmatrix} \qquad W^3 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Forward propagation :

- $x^0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ i.e (x = 2 and y = 1)
- $s^1 = (W^1)^T x^0 = \begin{bmatrix} 0.7 \\ 1 \end{bmatrix}$ and $x^1 = \begin{bmatrix} 1 \\ 0.6 \\ 0.76 \end{bmatrix}$
- $s^2 = (W^2)^T x^1 = \begin{bmatrix} -1.48 \end{bmatrix}$ and $x^2 = \begin{bmatrix} 1 \\ -0.9 \end{bmatrix}$
- $s^3 = (W^3)^T x^2 = \begin{bmatrix} -0.8 \end{bmatrix}$ and $x^3 = -0.8$

We know that $\delta^{(L)} = 2(x^{(L)} - y).\theta'(s^{(L)})$. As output transformation is linear $\theta'(s^{(L)}) = 1$. Therefore $\delta^{(L)} = 2(x^{(L)} - y)$.
$\Rightarrow \delta^3 = 2(x^3 - y) = 2(-0.8 - 1) = -3.6$

Backpropagation :

- $\delta^{(l)} = \theta'(s^{(l)}) \times [W^{(l+1)}.\delta^{(l+1)}]_1^{d(l)}$ and using **tanh** activation $\theta'(s^l) = [1 - x^l \times x^l]_1^{d(l)}$
- For s^2
$$\theta'(s^2) = 1 - (0.9)^2 = 0.19$$
$$\delta^2 = (0.19)[\begin{bmatrix} 1 \\ 2 \end{bmatrix} * (-3.6)]_1^1 = (0.19)(2 * -3.6) = -1.368$$
- For s^1
$$\theta'(s^1) = 1 - \begin{bmatrix} 0.6^2 \\ 0.76^2 \end{bmatrix} = \begin{bmatrix} 0.64 \\ 0.43 \end{bmatrix}$$
$$\delta^1 = \begin{bmatrix} 0.64 \\ 0.43 \end{bmatrix} \begin{bmatrix} 0.2 \\ 1 \\ -3 \end{bmatrix}_1^2 * -1.368 = \begin{bmatrix} 0.64 \\ 0.43 \end{bmatrix} \times \begin{bmatrix} -1.368 \\ 4.104 \end{bmatrix} = \begin{bmatrix} -0.87 \\ 1.76 \end{bmatrix}$$

Computing Gradients :

- $\frac{de}{dw^l} = x^{(l-1)}(\delta^l)^T$
- $\frac{de}{dw^1} = x^0(\delta^1)^T = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} -0.84 & 1.76 \end{bmatrix} = \begin{bmatrix} -0.87 & 1.76 \\ -1.74 & 3.52 \end{bmatrix}$
- $\frac{de}{dw^2} = x^1(\delta^2)^T = \begin{bmatrix} 1 \\ 0.6 \\ 0.76 \end{bmatrix} \begin{bmatrix} -1.368 \end{bmatrix} = \begin{bmatrix} -1.368 \\ -0.82 \\ -1.039 \end{bmatrix}$
- $\frac{de}{dw^3} = x^2(\delta^3)^T = \begin{bmatrix} 1 \\ -0.9 \end{bmatrix} \begin{bmatrix} -3.6 \end{bmatrix} = \begin{bmatrix} -3.6 \\ 3.24 \end{bmatrix}$

3

Ans. For calculating the no. of training parameters, let's assume the kernel size as (3, 3).

Standard Block

- Standard block has 2 convolution layers and 2 batch normalization layers. The input dimension of the image is $16 \times 16 \times 32$.
- For each convolution layer we have 32 kernels each of size $3 \times 3 \times 32$ and 1 bias term. The total parameters in each layer are $(3 \times 3 \times 32 + 1) \times 32 = 9248$.
- For each batch normalization layer, trainable parameters would be 2 times feature maps. The total parameters in each layer are $2 \times 32 = 64$
- Therefore, the number of parameters is $2 \times 9248 + 2 \times 64 = 18624$

Bottleneck Block

- Bottleneck block has 3 convolution layers and 3 batch normalization layers. The input dimension of the image is $16 \times 16 \times 128$.
- First convolution layer has 32 kernels each of size $1 \times 1 \times 128$ and 1 bias term. The total parameters in this layer is $(1 \times 1 \times 128 + 1) \times 32 = 4128$
- Next we have a batch normalization layer with 32 input feature maps. Trainable parameters are 2 times the number of feature maps. The total parameters in this layer is $2 \times 32 = 64$
- Second convolution layer has 32 kernels each of size $3 \times 3 \times 32$ and 1 bias term. The total parameters in this layer is $(3 \times 3 \times 32 + 1) \times 32 = 9248$
- Next we have a batch normalization layer with 32 input feature maps. The total parameters in this layer is $2 \times 32 = 64$
- Third convolution layer has 128 kernels each of size $1 \times 1 \times 32$ and 1 bias term. The total parameters in this layer is $(1 \times 1 \times 32 + 1) \times 128 = 4224$
- Next we have a batch normalization layer with 128 input feature maps. The total parameters in this layer is $2 \times 128 = 256$
- Therefore, the total number of parameters is $4128 + 64 + 9248 + 64 + 4224 + 256 = 17984$

Comparison of SB and BB

- In standard block we have both the 2 convolution layers with kernel size (3, 3) but in bottleneck block we have 3 convolution layers out of which one layer has kernel size of (3,3) and the other 2 with (1,1).
- Because of these 1×1 convolution layers, even if we have same number of input filters the number of trainable parameters for bottleneck block will be significantly lower compared to standard block as seen above.
- Due to lower number of parameters, bottleneck has an advantage over standard block. Bottleneck block requires less computational power compared to standard block and also avoids overfitting.
- As we have more layers in bottleneck architecture, it increases the model non-linearity and could easily fit more complex data.
- One disadvantage with bottleneck is, it uses identity convolution. Hence we lose benefit on bigger kernel size.

4

Ans.
(a) Given, Shape of tensor x as $N \times C$. The shape of both mean and variance is $1 \times C$.
(b) Given, tensor x is output of 2d convolution layer with shape $N \times H \times W \times C$. The shape of both mean and variance is $1 \times 1 \times C$.

5

Ans.

a) Given,

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \end{bmatrix} \in R^{2\times 4} \quad Y = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \in R^{2\times 2}$$
$$W^{ij} = \left[w_1^{ij}, w_2^{ij}, w_3^{i\cdot j}\right]$$
$$i = 1, 2, j = 1, 2.$$

where w^{ij} scans i-th channel of inputs and contributes to j-th channel of outputs.

- Calculating $\tilde{Y}$
- Y^{11}

$$Y^{11} = \begin{bmatrix} w_1^{11} & w_2^{11} & w_3^{11} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \end{bmatrix} + \begin{bmatrix} w_1^{21} & w_2^{21} & w_3^{21} \end{bmatrix} \begin{bmatrix} x_{21} \\ x_{22} \\ x_{23} \end{bmatrix}$$
$$Y^{11} = w_1^{11}x_{11} + w_2^{11}x_{12} + w_3^{11}x_{13} + w_1^{21}x_{21} + w_2^{21}x_{21} + w_3^{21}x_{21}$$

- Similarly Y^{12}

$$Y^{12} = w_1^{11}x_{12} + w_2^{11}x_{13} + w_3^{11}x_{14} + w_1^{21}x_{22} + w_2^{21}x_{23} + w_3^{21}x_{24}$$

- Similarly Y^{21}

$$Y^{21} = w_1^{12}x_{11} + w_2^{22}x_{12} + w_3^{42}x_{13} + w_1^{22}x_{12} + w_2^{22}x_{13} + w_3^{22}x_{14}$$

- Similarly Y^{21}

$$Y^{22} = w_1^{12}x_{12} + w_2^{22}x_{13} + w_3^{k2}x_{14} + w_1^{22}x_{22} + w_2^{22}x_{23} + w_3^{22}x_{24}$$

- Using above calculations $\tilde{Y}$ can be written as

$$\tilde{Y} = \underbrace{\begin{bmatrix} w_1^{11} & w_2^{11} & w_3^{11} & 0 & w_1^{21} & w_2^{21} & w_3^{21} & 0 \\ 0 & w_1^{11} & w_2^{11} & w_3^{11} & 0 & w_1^{21} & w_2^{21} & w_3^{21} \\ w_1^{12} & w_2^{12} & w_3^{12} & 0 & w_1^{22} & w_2^{22} & w_3^{22} & 0 \\ 0 & w_1^{12} & w_2^{12} & w_3^{12} & 0 & w_1^{22} & w_2^{22} & w_3^{22} \end{bmatrix}}_{\downarrow \atop A} \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \end{bmatrix}$$

b) Let us first write gradient for x_{11} and then we can generalize from it:

$$
\begin{aligned}
\frac{\partial L}{\partial x_{11}} &= \frac{\partial L}{\partial y_{11}}\frac{\partial y_{11}}{\partial x_{11}} + \frac{\partial L}{\partial y_{12}}\frac{\partial y_{12}}{\partial x_{11}} + \frac{\partial L}{\partial y_{21}}\frac{\partial y_{21}}{\partial x_{11}} + \frac{\partial L}{\partial y_{22}}\frac{\partial y_{22}}{\partial x_{11}} \\
&= \begin{bmatrix} \frac{\partial y_{11}}{\partial x_{11}} & \frac{\partial y_{12}}{\partial x_{11}} & \frac{\partial y_{21}}{\partial x_{11}} & \frac{\partial y_{22}}{\partial x_{11}} \end{bmatrix} \begin{bmatrix} \frac{\partial L}{\partial y^1_{11}} \\ \frac{\partial y_1}{\partial y_1} \\ \frac{\partial L}{\partial y_{21}} \\ \frac{\partial L}{\partial y_{22}} \end{bmatrix} \\
&= \begin{bmatrix} w_1^{11} & 0 & w_1^{12} & 0 \end{bmatrix} \frac{\partial L}{\partial \bar{Y}}
\end{aligned}
$$

If we write out the full matrix we would get the following:

$$
\frac{\partial L}{\partial \ddot{X}} = \begin{bmatrix}
w_1^{11} & 0 & w_1^{12} & 0 \\
w_2^{11} & w_1^{11} & w_2^{12} & w_1^{12} \\
w_3^{11} & w_2^{11} & w_3^{12} & w_2^{12} \\
0 & w_3^{11} & 0 & w_3^{12} \\
w_1^{21} & 0 & w_1^{22} & 0 \\
w_2^{21} & w_1^{21} & w_2^{22} & w_1^{22} \\
w_3^{21} & w_2^{21} & w_3^{22} & w_2^{22} \\
0 & w_3^{21} & 0 & w_3^{22}
\end{bmatrix} \frac{\partial L}{\partial \ddot{Y}}
$$

We can observe that $B = A^T$

c) Yes, $\frac{\partial L}{\partial X} = B\frac{\partial L}{\partial Y}$ can be considered as convolution on $\frac{\partial L}{\partial Y}$ to obtain $\frac{\partial L}{\partial X}$

Lets pad $\frac{\partial L}{\partial Y}$ on both sides

$$
\frac{\partial L}{\partial Y} = \begin{bmatrix} 0 & 0 & \partial L/\partial Y_{11} & \partial L/\partial Y_{12} & 0 & 0 \\ 0 & 0 & \partial L/\partial Y_{21} & \partial L/\partial Y_{22} & 0 & 0 \end{bmatrix}
$$

Let kernel be

$$
W = \left[W_3^{ij}, W_2^{ij}, W_1^{ij}\right] \quad \begin{matrix} i = 1, 2 \\ j = 1, 2 \end{matrix}
$$

$\Rightarrow$ After running convolution, The element at first row first column of output would be

$$
\begin{aligned}
&= 0 + 0 + w_1^{11}\frac{\partial L}{\partial Y_{11}} + 0 + 0 + w_1^{21}\frac{\partial L}{\partial Y_{21}} \\
&= w_1^{11}\frac{\partial L}{\partial Y_{11}} + w_1^{21}\frac{\partial L}{\partial Y_{21}}.
\end{aligned}
$$

which is the current $\frac{\partial L}{\partial X_{11}}$, Similarly we can get all elements of $\frac{\partial L}{\partial X}$.
Therefore, it is possible to view the gradient computation as a convolution on $\frac{\partial L}{\partial Y}$ to obtain $\frac{\partial L}{\partial X}$, with the kernel as following -

$$
W = \left[W_3^{ij}, W_2^{ij}, W_1^{ij}\right] \quad \begin{matrix} i = 1, 2 \\ j = 1, 2 \end{matrix}
$$

where W^{ij} scans the i-th channel from padded $\dfrac{\partial L}{\partial Y}$ and produces j-th channel in $\dfrac{\partial L}{\partial X}$