CIS 419/519: Homework 5

{Yupeng Li}

03.17.2020

Although the solutions are entirely my own, I consulted with the following people and sources while working on this homework: **https://www.youtube.com/watch?v=kNPGXgzxoHw**

PART I: PROBELM SET

1 Logical Functions with Neural Nets

(a) The NAND logic follows a truth table as follows:

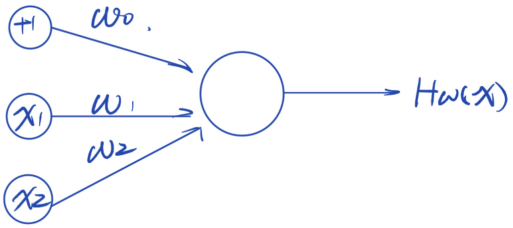
x_0	x_1	$H(x)$
0	0	1
0	1	1
1	0	1
1	1	0

If we use sigmoid as our activation function at the output node, where $\sigma(z) = \frac{1}{1+exp(-z)}$ and $H(x) = \sigma(w_0 + w_1x_0 + w_2x_2)$, where $w_0 = 40$, and $w_1 = w_2 = -25$ we would then have:

x_0	x_1	$H(x)$
0	0	$\sigma(40) = 1$
0	1	$\sigma(15) = 1$
1	0	$\sigma(15) = 1$
1	1	$\sigma(-10) = 0$

The neural network that is used to compute this function is the same as in the graph given in the problem set.

Figure 1: Neural network for NAND Logic

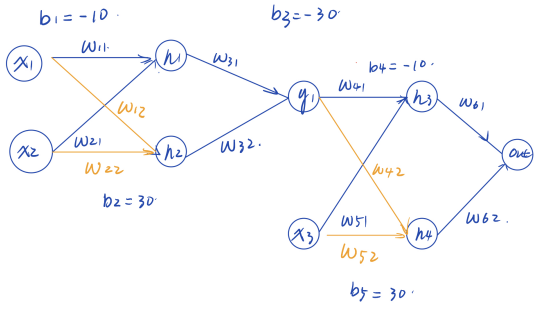


(b) The parity of three binary inputs follows a truth table as in the table underneath:

x_0	x_1	x_2	$H(x)$
1	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	0
1	1	1	1

The parity of three binary inputs can be characterized by two consecutive XOR gates. Neural network for the parity is demonstrated in the following figure:

Figure 2: Neural network for parity Logic



As in the figure, the weights are

$$w_{11} = w_{21} = w_{41} = w_{51} = 20$$

$$w_{12} = w_{22} = w_{42} = w_{52} = -20$$

$$w_{31} = w_{32} = w_{61} = w_{62} = 20$$

$$b_1 = b_4 = -10$$

$$b_2 = b_5 = 30$$

Sigmoid is applied as activation function in each hidden layer. Thus the left half can easily be proved to have XOR logic. Combining two XOR gates we can easily get the truth table as above.
For this section of homework, I referred to **https://www.youtube.com/watch?v=kNPGXgzxoHw**

2 Calculating Backprop by Hand

We can first calculate the hidden layer using the following equation:

$$Sign(\begin{bmatrix} 0.1 & 0.2 \\ -0.4 & 0.3 \end{bmatrix} \begin{bmatrix} 5 \\ 4 \end{bmatrix}) = H \tag{1}$$

Thus,

$$H = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{2}$$

The output is then

$$\sigma(\begin{bmatrix} 0.1 & 0.2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix}) = 0.524979 \tag{3}$$

The final output can be written in the form

$$Output = \sigma(1 \cdot W_1^2 - 1 \cdot W_2^2) = \frac{1}{1 + e^{-(1 \cdot W_1^2 - 1 \cdot W_2^2)}}$$

Thus,

$$Output = \frac{1}{1 + e^{W_2^2 - W_1^2}}$$

Taking partial derivative with respect to W_2^1 and W_2^2 respectively, we can easily obtain the gradient matrix between the hidden layer and output layer which is:

$$h = \begin{bmatrix} 0.249376 \\ -0.249376 \end{bmatrix} \tag{4}$$

For the gradient with respect to the input layer:

$$W^1 = \begin{bmatrix} 0.1 & 0.2 \\ -0.4 & 0.3 \end{bmatrix} \tag{5}$$

The gradient of the output with respect to the top left weight W_{11}^1 is just:

$$\frac{\partial h(1)}{\partial W_{11}^1} \times \frac{\partial Output}{\partial h(1)}$$

Based on calculations:

$$\frac{\partial Output}{\partial h(1)} = 0.0249376$$

$$\frac{\partial h(1)}{\partial W_{11}^1} = 5 \cdot \nabla Sign(1) = 5$$

Thus the output gradient with respect to W_{11}^1 is just 0.124688

Following similar reasoning, the gradient between the output and the first layer weight can be constructed as in the following matrix:

$$\begin{bmatrix} 0.124688 & 0.249376 \\ 0.0997504 & 0.199501 \end{bmatrix} \tag{6}$$

PART II: PROGRAMMING EXERCISES All of part 2 are submitted in HW5.ipynb and cnn.th