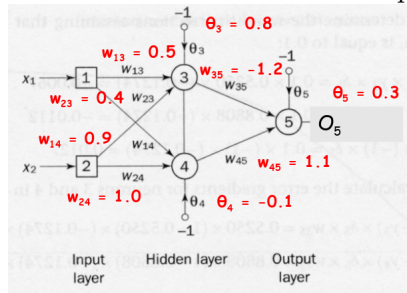


# COMP 472 Artificial Intelligence (Winter 2024)

## Worksheet #5: Introduction to Deep Learning

**Matrix Notation.** Last time, we labeled each weight in a neural network individually ( $w_{14}, w_{45}, \dots$ ), which is not practical for larger networks. From now on, we'll use *matrix notation*:<sup>1</sup> We can encode the sample input  $x$  as a matrix of size  $1 \times 2$ , the weights  $W$  as a matrix of size  $2 \times 2$  and the bias  $b$  as a matrix of size  $1 \times 2$ . We can then compute the net activation as the dot product and add the bias:  $\text{net} = x \cdot W + b$ :



$$\text{net}_h = \begin{bmatrix} 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.5 & 0.9 \\ 0.4 & 1.0 \end{bmatrix} + \begin{bmatrix} -0.8 & 0.1 \end{bmatrix} = \begin{bmatrix} . & . & . & . & . \end{bmatrix}$$

(Next, you'd have to compute the activation function to get the output:  $h = \text{sigmoid}(\text{net}_h)$ , as before; and finally do the same calculation for the output layer.)

**Autoencoder.** Assume that the  $3 \times 3$  matrix below represents a gray scale image:

$$X = \begin{bmatrix} 0.2 & 0.3 & 0.2 \\ 0.4 & 0.1 & 0.3 \\ 0.1 & 0.9 & 0.5 \end{bmatrix}$$

Your job is to use  $X$  to train an *autoencoder*. So, the input to our network is going to be  $X$ . But what is the expected output (labels)? .....

Define the size of the input vector: ..... and output vector: ..... of your autoencoder.

How many hidden neurons  $H$  would you use? (Assuming we use a single hidden layer): .....  
(Hint: there is no single correct answer, but you can define a sensible range.)

How many *weights* are there to train in total? (For  $H$  hidden neurons): .....

**Autoencoder Activation.** Assume that we use an autoencoder with the following hyperparameters: the activation function is *sigmoid* and the hidden layer has a size of 5. Perform a single forward pass through the autoencoder. You can assume an input value of 1 for the biases, and all the weights (including the biases) are initialized to 0.5. Note, our input vector corresponding to the image above is

$$x = \begin{bmatrix} 0.2 & 0.3 & 0.2 & 0.4 & 0.1 & 0.3 & 0.1 & 0.9 & 0.5 \end{bmatrix}$$

1. First, compute the pre-activation function (the *net*), by multiplying the input and weights, plus the bias  $b$ . Note that  $W$  is a matrix of size  $9 \times 5$ :

$$\text{net}_h = x \cdot W_{ih} + b_{ih} \quad (1)$$

the result is a matrix of size  $1 \times 5$ :  $\text{net}_h = [ \dots ]$

<sup>1</sup>See <https://medium.com/from-the-scratch/deep-learning-deep-guide-for-all-your-matrix-dimensions-and-calculations-415012de1568> for a review of matrix calculations for neural networks

2. Now, to compute the result for  $h$ , the sigmoid function  $S(x) = \frac{1}{1+e^{-x}}$  is applied to the pre-activation (net) result (eq. 1):

$$h = S(\text{net}_h) \quad (2)$$

where  $h$  is again a matrix of size  $1 \times 5$ :  $h = [ \dots ]$

3. To compute the output  $o$ , the result of the hidden layer (eq. 2) should be multiplied with the weights of the output layer,  $W_{ho}$ , then we apply sigmoid again:

$$o = S(h \cdot W_{ho} + b_{ho}) \quad (3)$$

where  $o$  is now a matrix of size  $1 \times 9$ :  $o = [ \dots ]$

**CNN Activation Map.** Assume the following matrix that represents an image. This image will be fed to a convolutional neural network (CNN).

1	1	2	2	2	0	0
2	0	1	1	2	1	2
0	1	0	0	1	1	2
0	2	1	2	0	2	2
1	2	0	0	1	0	1
0	0	0	0	1	2	1
2	0	0	0	2	1	1

Assume that we use the following convolution filter (kernel) with a stride of 2 (no padding):

0	1	1
0	1	0
0	-1	-1

- What will be the size of the activation map? .....
- What will be the resulting activation map?

**Pooling Layer.** What will be the output of the pooling layer with a size of  $2 \times 2$  and a stride of 1, on the activation map of the question above, if we use the following strategies:

- Average pooling:
- Max pooling: