Deep Learning
Tutorial 1: Signals and Convolution

Q1: The curse of dimensionality

1a) For a one-dimensional space of real numbers between 0 and 1, 100 observations are required to adequately cover this space such that histograms can be calculated and conclusions can be drawn. How many observations would be required to adequately cover the space for the following number of dimensions?

i) 10 dimensions

ii) 1000 dimensions

1b) Under what conditions is it still possible to perform Cluster analysis and outlier detection in high-dimensional spaces?

1c) Why is the curse of dimensionality a serious hurdle in machine learning problems?

1d) Why is using weight sharing common practice?

1	1	2	3	2	1	1
1	2	6	6	5	2	1
1	5	6	7	7	6	2
1	5	6	6	6	4	2
1	1	4	5	4	1	1
1	1	3	4	3	1	0
0	0	1	2	1	0	1

(a) Single channel image

0	1	0
0	-2	1
0	1	0

(b) 3x3 Filter

Figure 1: Perform a convolution

Q2: Convolutions

2a) Suppose you have the following single-channel image shown in Figure 1a:

i) Compute the result of max pooling of size 3x3 with stride=2.

ii) Compute the result of convolution of the input image with the 3x3 filter shown in Figure 1b using stride=2, no zero padding.

2bi) How would the filter shown in Figure 1b need to changed to obtain the correct definition of convolution rather than cross-correlation? Check your answer to **2aii)**.

2bii) Why do we need to perform this change?

2c) Which of the following properties hold for convolution? (True/False)

i) Non-Commutativity: $f * g \neq g * f$

ii) Associativity: $f * (g * h) = (f * g) * h$

iii) Non-Distributivity: $f * (g + h) \neq (f * g) + (f * h)$

iv) Associativity with scalar multiplication: $a(f * g) = (af) * g$

v) Derivative: $D(f * g) = (Df) * g = f * Dg$

Q3: Backpropagation applied

A Siamese network is a type of neural network architecture that consists of two or more identical subnetworks (called "twins") that share the same weights and architecture. Siamese networks are often used for tasks that involve comparing or matching inputs, such as verification, identification, and similarity learning. The outputs of the twin networks are usually joined later on by more layers. Let's assume we have a two layer Siamese neural network, as defined below:

$$
\begin{aligned}
z_1 &= W_1 x^{(i)} + b_1 \\
a_1 &= ReLU(z_1) \\
z_2 &= W_1 x'^{(i)} + b_1 \\
a_2 &= ReLU(z_2) \\
a &= a_1 - a_2 \\
z_3 &= W_2 a + b_2 \\
\hat{y}^{(i)} &= \sigma(z_3) \\
\mathcal{L}^{(i)} &= y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \\
J &= -\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}^{(i)}
\end{aligned}
$$

Note that ($x^{(i)}$, $x'^{(i)}$) represents a pair of single input examples, and are each of shape $D \times 1$. Further $y^{(i)}$ is a single output label and is a scalar. There are m examples in our dataset. We use D_{a1} nodes in our first hidden layers; *i.e.*, z_1's and z_2's shape is $D_{a1} \times 1$. Note that the first two layers share the same weights.

1. What are the shapes of W_1, b_1, W_2, b_2? If we were vectorizing across multiple examples, what would the shapes of X and Y be instead?
2. Derive $\frac{\partial J}{\partial z_3}$ formally and write $\delta_1^i = ...$ You can simplify the equation in terms of $\hat{y}^{(i)}$.
3. Derive $\frac{\partial z_3}{\partial a}$ formally and write $\delta_2^i = ...$
4. Derive $\frac{\partial a}{\partial z_2}$ formally and write $\delta_3^i = ...$