TDT4195: Visual Computing Fundamentals

Image Processing - Assignment 1 Report

27th October 2023

Oluwatobi OJEKANMI

1 Spatial Filtering Theory

a.

Sampling is the process of spatially converting a continuous signal (analog image) into a discrete form (digital image) by determining which portion of the continuous signal corresponds to each pixel in the digital image.

b.

Quantization is the process of converting a continuous range of intensity values into a discrete and finite set of values that a fixed number of bits can represent.

c.

For a non-binary intensity image, increasing the contrast of the image means decreasing low-intensity values and increasing high-intensity values (i.e., making lighter pixels lighter and dark pixels darker). Therefore, such an image has high contrast if the image histogram displays a non-uniform spread of intensity values such that the low-intensity and high-intensity values have high distributions while other intensity values have low distribution values.
The higher the peaks of the lowest and highest intensity values compared to other intensity values, the higher the image contrast.

d.

i. Find the histogram of the input image intensity values, $H_r(r_k) = n_k$

r_k	0	1	2	3	4	5	6	7
$H_r(r_k)$	1	1	0	1	2	2	4	4

ii. Determine the probability density function or PDF of the distribution, $p_r(r_k)$

$$p_r(r_k) = \frac{n_k}{M \cdot N}$$

r_k	0	1	2	3	4	5	6	7
$H_r(r_k)$	1	1	0	1	2	2	4	4
$p_r(r_k)$	$\frac{1}{15}$	$\frac{1}{15}$	0	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{4}{15}$	$\frac{4}{15}$

iii. Find the cumulative density function or CDF of the distribution, $F_r(r_k)$

$$F_r(r_k) = \sum_{j=0}^{k} p_r(r_j)$$

r_k	0	1	2	3	4	5	6	7
$H_r(r_k)$	1	1	0	1	2	2	4	4
$p_r(r_k)$	$\frac{1}{15}$	$\frac{1}{15}$	0	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{4}{15}$	$\frac{4}{15}$
$F_r(r_k)$	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{3}{15}$	$\frac{5}{15}$	$\frac{7}{15}$	$\frac{11}{15}$	$\frac{15}{15}$

iv. Find and scale to the desired range of intensity levels and round down (floor) output using the intensity transform function $T(r)$

$$\begin{aligned} T(r) &= (L-1) \cdot F_r(r) \\ &= 7 \cdot F_r(r_k) \end{aligned}$$

r_k	0	1	2	3	4	5	6	7
$H_r(r_k)$	1	1	0	1	2	2	4	4
$p_r(r_k)$	$\frac{1}{15}$	$\frac{1}{15}$	0	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{4}{15}$	$\frac{4}{15}$
$F_r(r_k)$	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{3}{15}$	$\frac{5}{15}$	$\frac{7}{15}$	$\frac{11}{15}$	$\frac{15}{15}$
$T_r(r_k)$	0	0	0	1	2	3	5	7

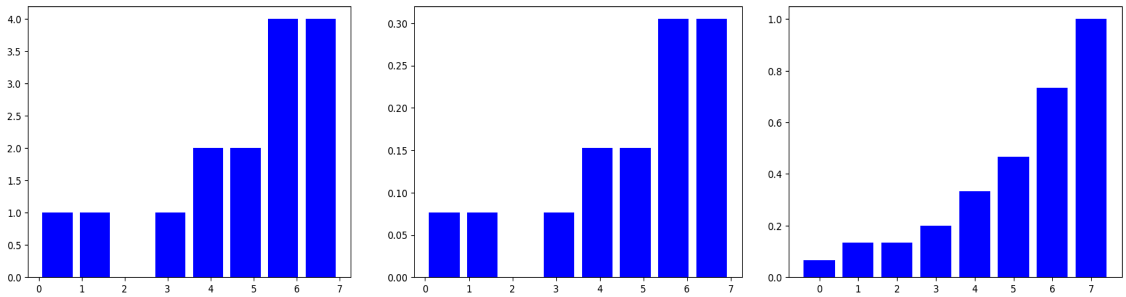
v. Map new intensity values to the image

Given Image				
1	7	6	3	6
7	6	5	6	4
5	4	7	7	0

$\rightarrow$

New Image				
0	7	5	1	5
7	5	3	5	2
3	2	7	7	0

Figure 1 shows the histogram, PDF, and CDF plots of the image's intensity distribution.



(a) Distribution's Histogram (b) Distribution's PDF (c) Distribution's CDF

Figure 1: Distribution's Histogram, PDF, and CDF plots

e.

Applying a log transform to an image with a large variance in pixel intensities will compress the dynamic range of the image. For example, applying a log transform to a digital image with intensity values ranging from 0 to 255 will compress and map the intensity values to a new range of 0 to $2.41c$, where c is the scaling factor. The choice of c determines the new intensity range.
Furthermore, using a scaling factor c that converts the new intensity range from $\{0, 2.41c\}$ to $\{0, 255\}$ changes darker intensity values to brighter values, thus making the details present in darker or gray areas of the image more visible to human eyes.

f.

i. Pad the original image with zeros

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

$\rightarrow$

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

ii. Flip the Kernel

1	0	-1
2	0	-2
1	0	-1

$\rightarrow$

-1	0	1
-2	0	2
-1	0	1

iii. Perform cross-correlation on the padded image with the flipped kernel. The result of this operation is

20	8	-8	-1	-12
23	3	-1	-9	-22
14	2	6	-15	-20

2 Spatial Filtering Programming

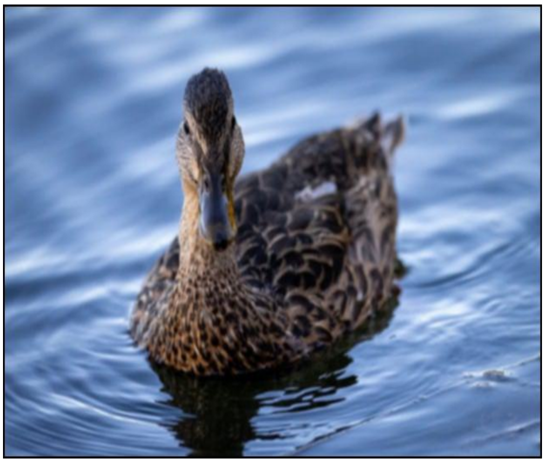
a.

Figure 2: duck.jpeg as a grayscale image.

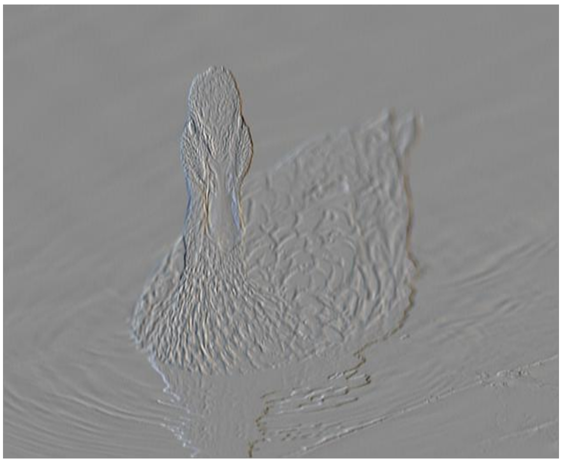
b.

Figure 3: duck.jpeg as a transformed grayscale image

c.



(a) Duck Convolved with Smoothing Kernel



(b) Duck Convolved with Sobel Kernel

Figure 4: Convolved Duck Images

Kindly note that I padded the output image from the convolution operation with zeros to get an image of the same size as the input image. For an odd $m \times n$ kernel, $(m-1,\, n-1)$ padding is required for both sides. Therefore, the image was padded on the top and bottom with $\frac{m-1}{2}$ rows and on the left and right with $\frac{n-1}{2}$ columns.

3 Neural Networks Theory

a.

XOR: XOR is not linearly separable. This implies that we cannot draw a single straight line, plane, or hyperplane to separate the XOR function's output of 0s and 1s. XOR requires a non-linear decision boundary to be represented accurately. Therefore, the XOR function cannot be represented by a single-layer neural network.

b.

A hyperparameter for a neural network is a configuration set prior to the training process and cannot be learned from the data. Examples of a hyperparameter include the batch size, learning rate, number of hidden layers, or number of neurons in each layer.

c.

The softmax activation function is used in the last layer of neural networks trained to classify objects because it converts raw output scores (logits) into probabilities, ensuring that the output values are between 0 and 1 and that they sum up to 1. This is crucial for multi-class classification problems, where the goal is to assign an input instance to one of several possible classes. Softmax function thus provides a probability distribution over the classes, making it easy to interpret the network's output as the likelihood of the input belonging to each class.
For binary classification tasks, the sigmoid activation function can also be used in the last layer instead of a softmax activation function.

d.

i. Run a forward pass

$$\begin{aligned}
a_1 &= w_1 \cdot x_1 = -1 \cdot -1 = \mathbf{1} \\
a_2 &= w_2 \cdot x_2 = 1 \cdot 0 = \mathbf{0} \\
a_3 &= w_3 \cdot x_3 = -1 \cdot -1 = \mathbf{1} \\
a_4 &= w_4 \cdot x_4 = -2 \cdot 2 = \textbf{-4} \\
c_1 &= a_1 + a_2 + b_1 = 1 + 0 + 1 = \mathbf{2} \\
c_2 &= a_3 + a_4 + b_2 = 1 - 4 - 1 = \textbf{-4} \\
\hat{y} &= max(c_1, c_2) = max(2, -4) = \mathbf{2} \\
C &= \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(1-2)^2 = \mathbf{\frac{1}{2}}
\end{aligned}$$

ii. Compute relative gradients

$$\begin{aligned}
&\frac{\partial C}{\partial \hat{y}} = -(y - \hat{y}) = -(1-2) = \mathbf{1} \\
&\frac{\partial \hat{y}}{\partial c_1} = \mathbf{1}; \quad \frac{\partial \hat{y}}{\partial c_2} = \mathbf{0} \\
&\frac{\partial c_1}{\partial a_1} = \mathbf{1}; \quad \frac{\partial c_1}{\partial a_2} = \mathbf{1}; \quad \frac{\partial c_1}{\partial b_1} = \mathbf{1} \\
&\frac{\partial c_2}{\partial a_3} = \mathbf{1}; \quad \frac{\partial c_2}{\partial a_4} = \mathbf{1}; \quad \frac{\partial c_2}{\partial b_2} = \mathbf{1} \\
&\frac{\partial a_1}{\partial w_1} = x_1 = \textbf{-1}; \quad \frac{\partial a_2}{\partial w_2} = x_2 = \mathbf{0}; \quad \frac{\partial a_3}{\partial w_3} = x_3 = \textbf{-1}; \quad \frac{\partial a_4}{\partial w_4} = x_4 = \mathbf{2};
\end{aligned}$$

iii. Compute the gradients of the loss relative to the model parameters

$$\frac{\partial C}{\partial w_1} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_1} \cdot \frac{\partial c_1}{\partial a_1} \cdot \frac{\partial a_1}{\partial w_1} = 1 \cdot 1 \cdot 1 \cdot -1 = \textbf{-1}$$
$$\frac{\partial C}{\partial w_2} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_1} \cdot \frac{\partial c_1}{\partial a_2} \cdot \frac{\partial a_2}{\partial w_2} = 1 \cdot 1 \cdot 1 \cdot 0 = \mathbf{0}$$
$$\frac{\partial C}{\partial w_3} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_2} \cdot \frac{\partial c_2}{\partial a_3} \cdot \frac{\partial a_3}{\partial w_3} = 1 \cdot 0 \cdot 1 \cdot -1 = \mathbf{0}$$
$$\frac{\partial C}{\partial w_4} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_2} \cdot \frac{\partial c_2}{\partial a_4} \cdot \frac{\partial a_4}{\partial w_4} = 1 \cdot 0 \cdot 1 \cdot 2 = \mathbf{0}$$
$$\frac{\partial C}{\partial b_1} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_1} \cdot \frac{\partial c_1}{\partial b_1} = 1 \cdot 1 \cdot 1 = \mathbf{1}$$
$$\frac{\partial C}{\partial b_2} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_2} \cdot \frac{\partial c_2}{\partial b_2} = 1 \cdot 0 \cdot 1 = \mathbf{0}$$

e.

$$w_1 := w_1 - \alpha \cdot \frac{\partial C}{\partial w_1} = -1 - (0.1 \cdot -1) = -1 + 0.1 = \textbf{-0.9}$$
$$w_3 := w_3 - \alpha \cdot \frac{\partial C}{\partial w_3} = -1 - (0.1 \cdot 0) = -1 - 0 = \textbf{-1}$$
$$b_1 := b_1 - \alpha \cdot \frac{\partial C}{\partial b_1} = 1 - (0.1 \cdot 1) = 1 - 0.1 = \mathbf{0.9}$$

4 Neural Networks Programming

a.

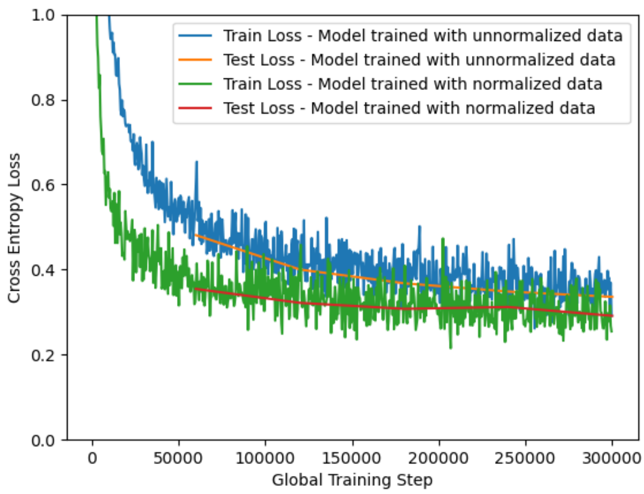


Figure 5: Train and Test Losses for Models with Normalized and Unnormalized Data

Figure 5 shows that the loss converges faster when the data is normalized. This implies that the model finds it easier to find a classification pattern in the normalized data as compared to when the data were not normalized.

b.

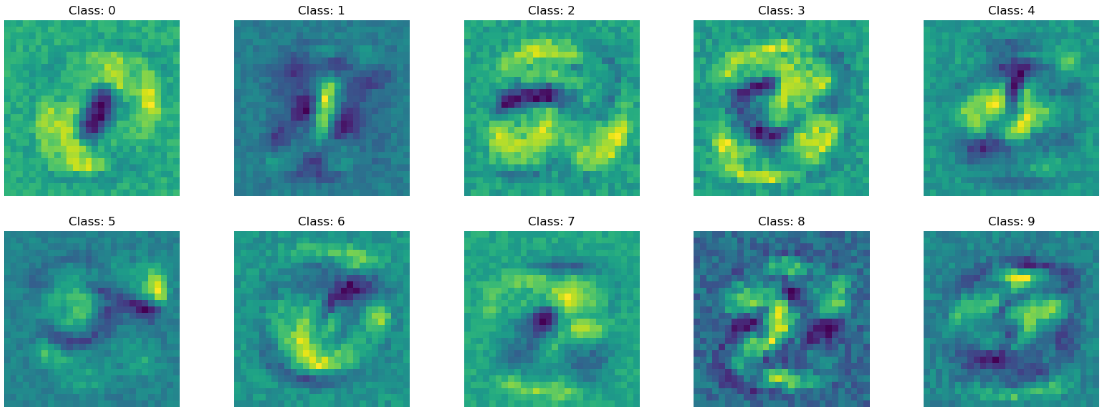


Figure 6: Model Weights Plots

Figure 6 shows that the model is learning relevant patterns/shapes to recognize and classify each digit. For example, the model weights for classes 0 and 3 show the shape/pattern of the original digit being detected/classified.

c.

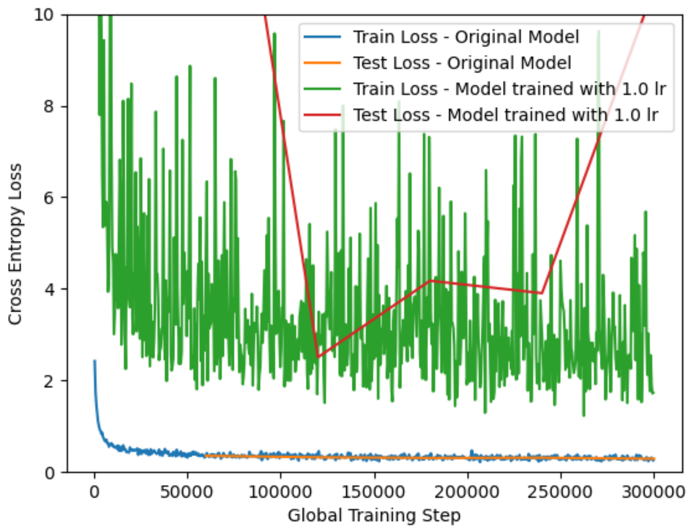


Figure 7: Train and Test Losses for Models with 0.0192 and 1.0 Learning Rates

Learning Rate	Validation Loss	Validation Accuracy
0.0192	0.29	0.92
1.0	10.56	0.72

Effect of different learning rates on loss convergence

Figure 7 shows that the loss function is finding it difficult to converge to local minima with a 1.0 learning rate. This is because the learning rate is high and thus, the loss oscillates around the minima. Reducing the learning rate would enable convergence.

d.

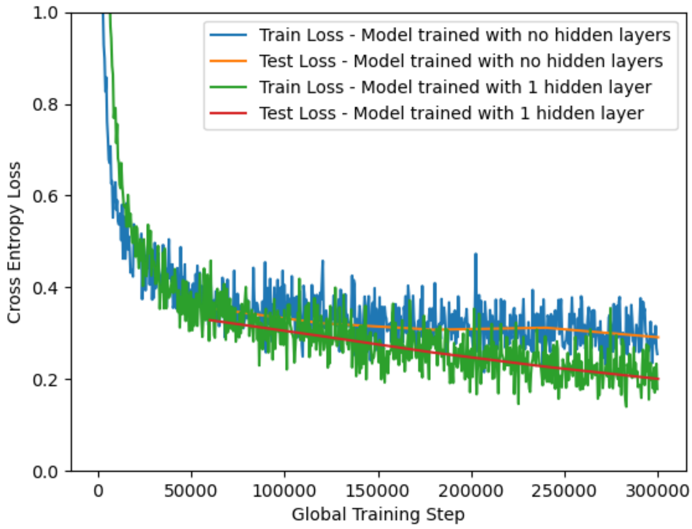


Figure 8: Train and Test Losses for Models with no and one hidden layer

Figure $\boxed{8}$ shows that the loss function for the model with a hidden layer converges faster than the loss of the model with no hidden layer. This is because adding hidden layers gives the model more capacity to learn more classification patterns.