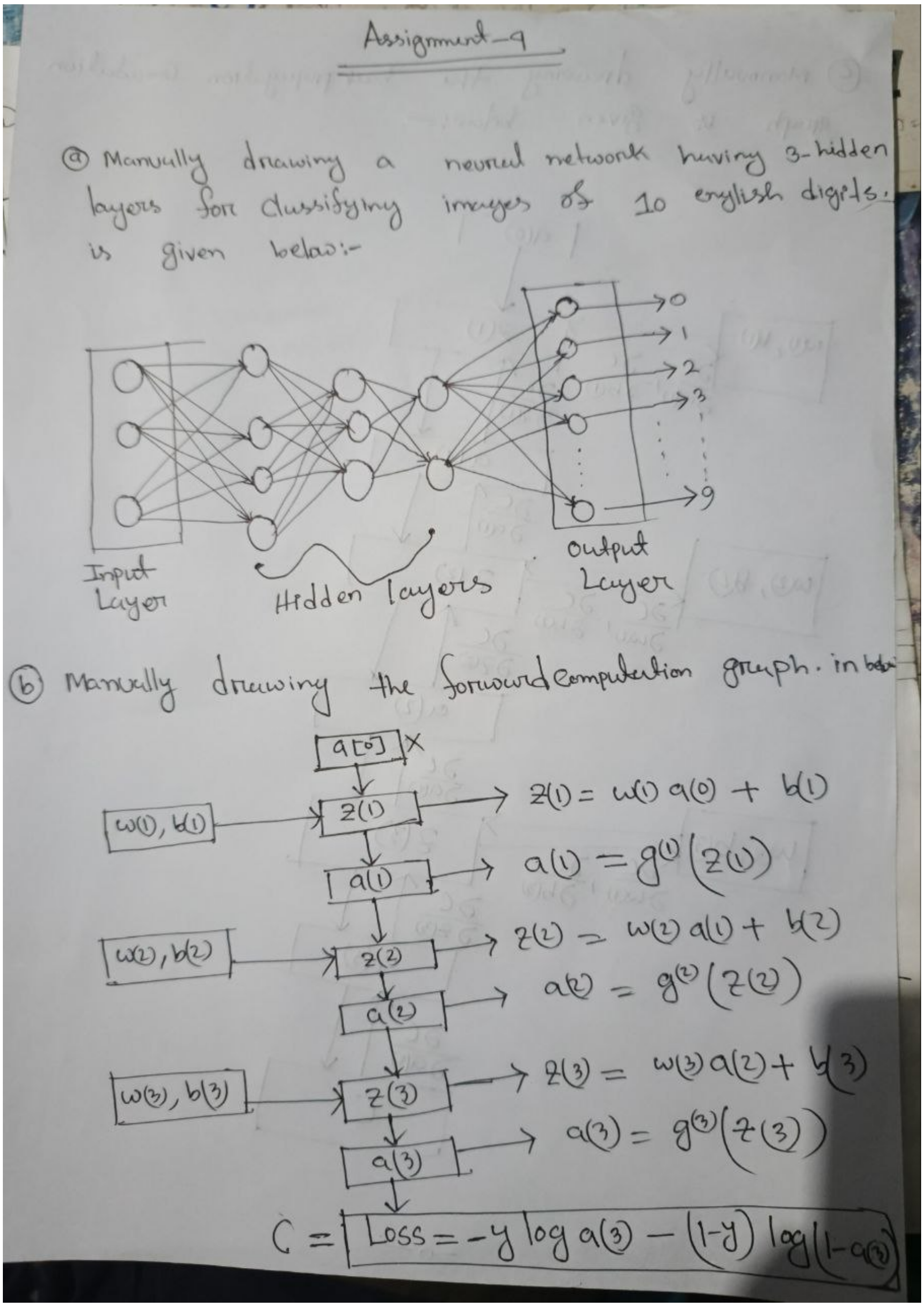
Md: Israil Hosen

Roll: 2010876110

Neural Network and Deep Learning Assignment-4

[Code link]

a & b



Assignment-9

ⓐ Manully drawing a neural network having 3-hidden layers for classifying images of 10 english digits. is given below:-

0
1
2
3
9
Input Layer
Hidden layers
Output Layer

ⓑ Manually drawing the forwardcomputation graph. in below

a[0] X
z(1)
w(1), b(1)
a(1)
w(2), b(2)
z(2)
a(2)
w(3), b(3)
z(3)
a(3)

$z(1) = w(1)\, a(0) + b(1)$

$a(1) = g^{(1)}(z(1))$

$z(2) = w(2)\, a(1) + b(2)$

$a(2) = g^{(2)}(z(2))$

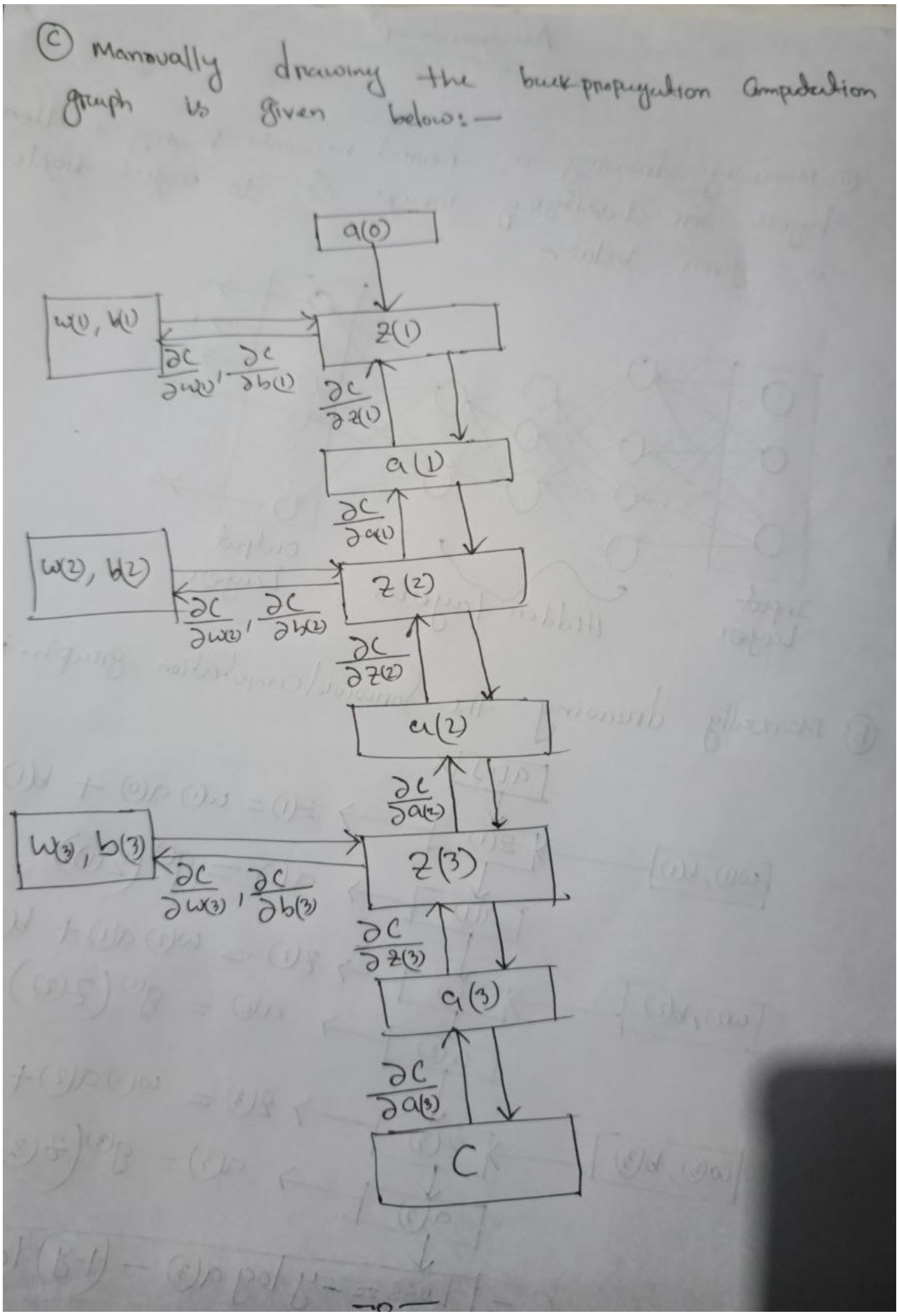
$z(3) = w(3)\, a(2) + b(3)$

$a(3) = g^{(3)}(z(3))$

$C = \boxed{\text{Loss} = -y \log a(3) - (1-y) \log(1-a(3))}$

Figure 1: Model Architecture and Forward Computation Graph

C

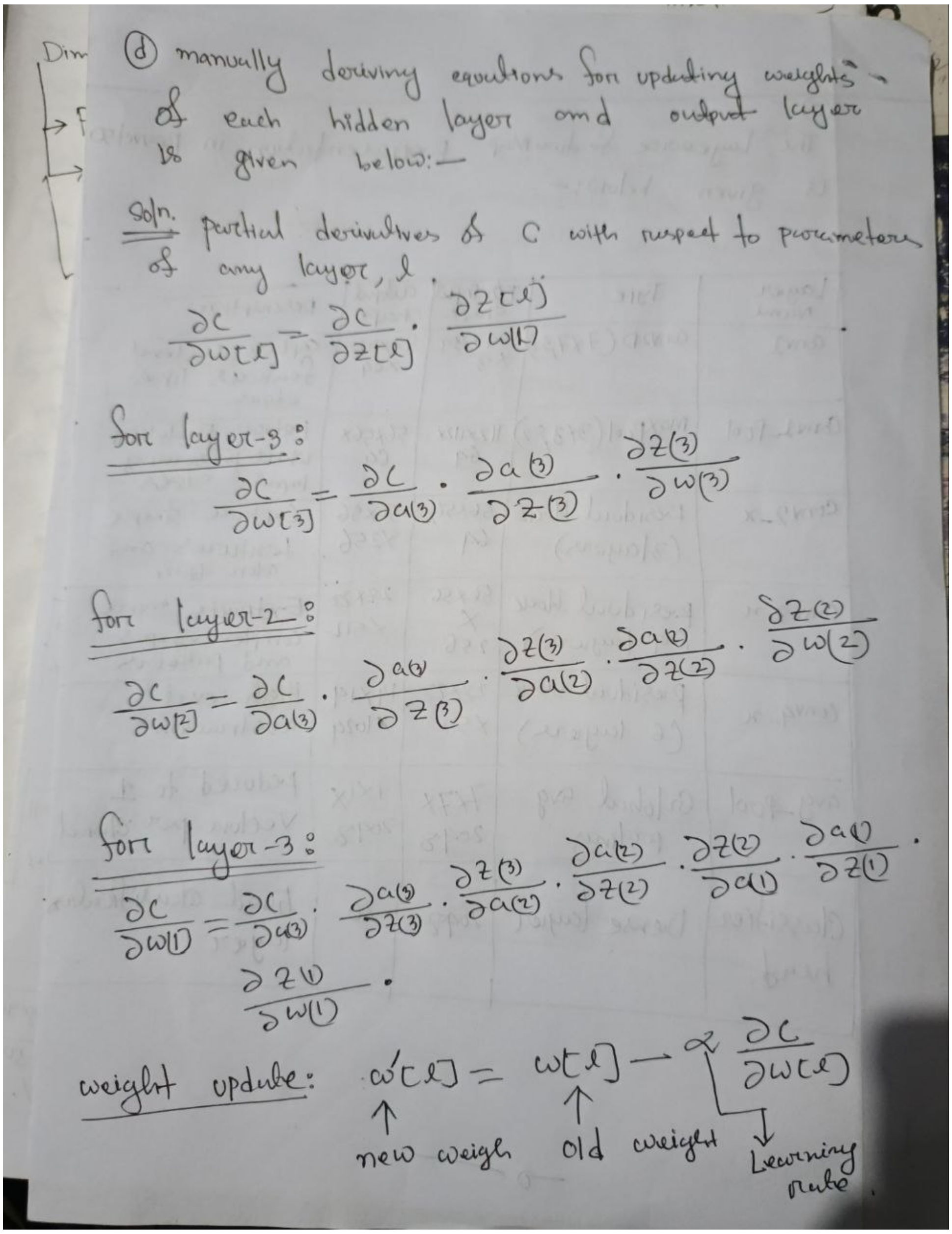


(c) Manually drawing the back-propagation computation graph is given below:—

$a(0)$
$w(1), b(1)$
$z(1)$
$\frac{\partial C}{\partial w(1)}, \frac{\partial C}{\partial b(1)}$
$\frac{\partial C}{\partial z(1)}$
$a(1)$
$\frac{\partial C}{\partial a(1)}$
$w(2), b(2)$
$z(2)$
$\frac{\partial C}{\partial w(2)}, \frac{\partial C}{\partial b(2)}$
$\frac{\partial C}{\partial z(2)}$
$a(2)$
$\frac{\partial C}{\partial a(2)}$
$w(3), b(3)$
$z(3)$
$\frac{\partial C}{\partial w(3)}, \frac{\partial C}{\partial b(3)}$
$\frac{\partial C}{\partial z(3)}$
$a(3)$
$\frac{\partial C}{\partial a(3)}$
C

Figure 2: Backpropagation Computation Graph

d



(d) manually deriving equations for updating weights of each hidden layer and output layer is given below:—

Soln. partial derivatives of C with respect to parameters of any layer, l.

$$\frac{\partial C}{\partial w[l]} = \frac{\partial C}{\partial z[l]} \cdot \frac{\partial z[l]}{\partial w[l]}.$$

for layer-3:

$$\frac{\partial C}{\partial w[3]} = \frac{\partial C}{\partial a(3)} \cdot \frac{\partial a(3)}{\partial z(3)} \cdot \frac{\partial z(3)}{\partial w(3)}$$

for layer-2:

$$\frac{\partial C}{\partial w[2]} = \frac{\partial C}{\partial a(3)} \cdot \frac{\partial a(3)}{\partial z(3)} \cdot \frac{\partial z(3)}{\partial a(2)} \cdot \frac{\partial a(2)}{\partial z(2)} \cdot \frac{\partial z(2)}{\partial w(2)}$$

for layer-3:

$$\frac{\partial C}{\partial w(1)} = \frac{\partial C}{\partial a(3)} \cdot \frac{\partial a(3)}{\partial z(3)} \cdot \frac{\partial z(3)}{\partial a(2)} \cdot \frac{\partial a(2)}{\partial z(2)} \cdot \frac{\partial z(2)}{\partial a(1)} \cdot \frac{\partial a(1)}{\partial z(1)} \cdot \frac{\partial z(1)}{\partial w(1)}.$$

weight update:

$$\underset{\substack{\uparrow \\ \text{new weigh}}}{w'[l]} = \underset{\substack{\uparrow \\ \text{old weight}}}{w[l]} - \underset{\substack{\downarrow \\ \text{Learning rate}}}{\alpha} \frac{\partial C}{\partial w[l]}$$

Figure 3: Derive equations for updating weights

e

The model will be trained on the MNIST digit dataset using TensorFlow's tf.GradientTape() API, which allows the training process to be handled with more control compared to using the high-level model.fit() method.
CodeLink: [**click here**]

f

Comparing the performance of the model trained by tf.GradientTape() with Tensorflow's model.fit() are given below:

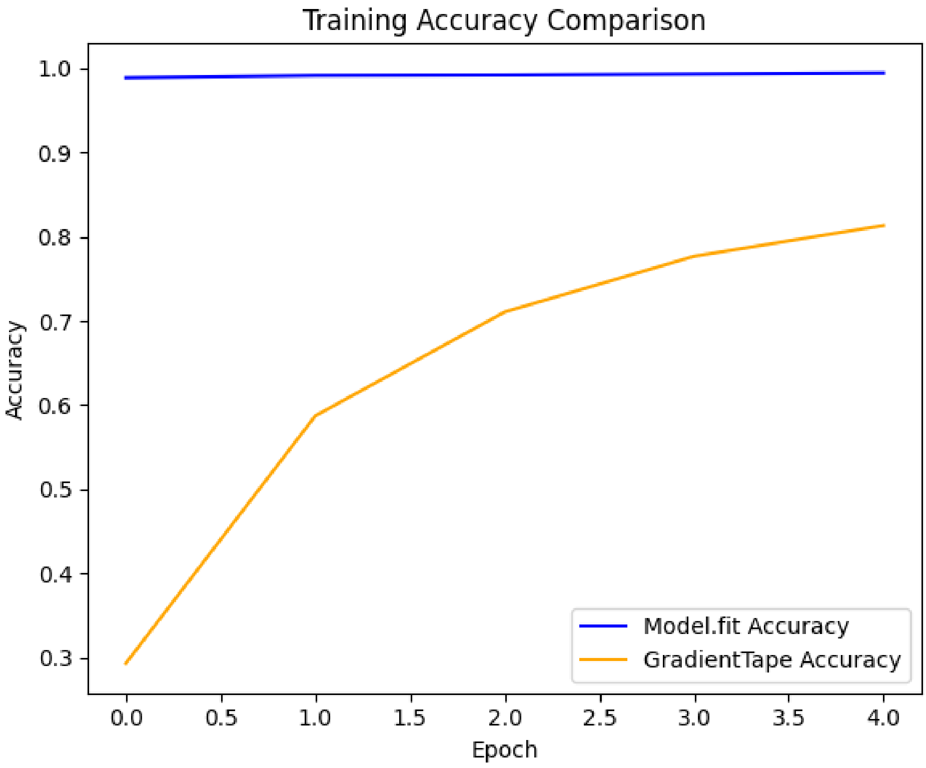


Figure 4: Training accuracy comparison

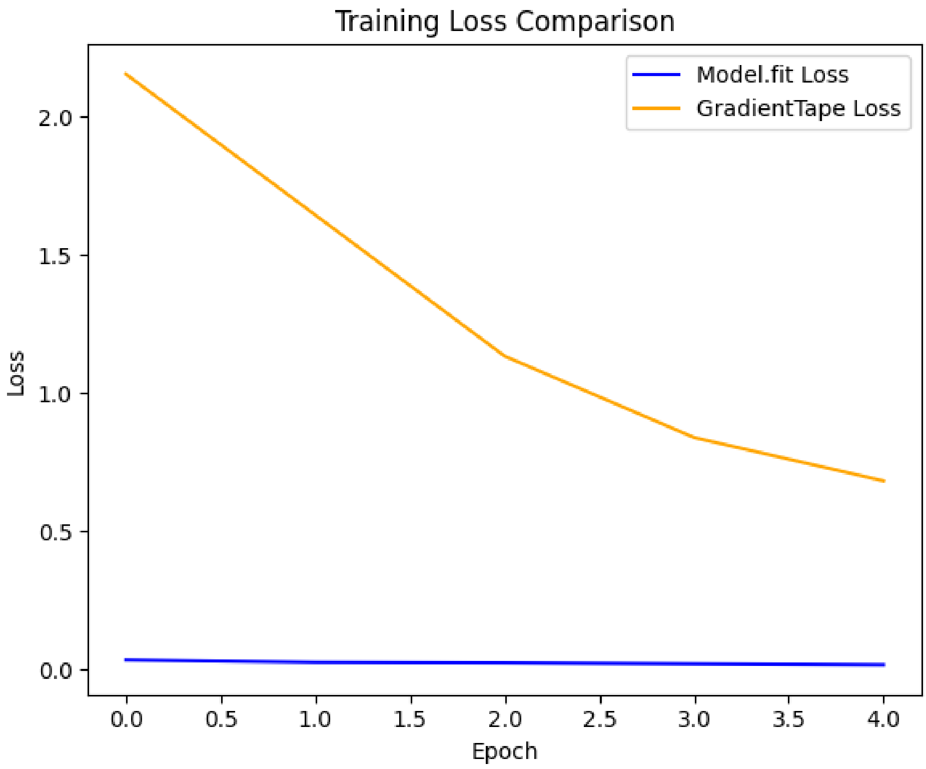


Figure 5: Training loss comparison

The test accuracy achieved using GradientTape is 82.97%, while the model trained with model.fit() achieved a higher test accuracy of 97.97%.