Deep Learning for Computer Vision (CS776A)
Indian Institute of Technology Kanpur
Assignment 1

1

Member Names: Aditya Loth
Member Emails: adityal21@iitk.ac.in
Member Roll Numbers: 21111004
Date: February 3, 2022

---

1. **Data Preprocessing:**
   The training data is reshaped to the (50000, 32, 32, 3) and testing data to (10000, 32, 32, 3)

2. **MLP model** The Neural network has input layer, single hidden layer with 64 neurons and an output layer.

   (a) **Input Layer:**
   Input is taken in batches of images where each image consist of 512 features. This makes the size of input as (batch size, 512).

   (b) **Hidden Layer:**
   The hidden layer has 64 neurons. It is implement using a randomly initialized weight matrix W1 and bias b1. It uses the input layer, W1 and bias to create a new matrix to be passed in output la

   (c) **Output Layer**
   It takes the output of hidden layer as input and convert them to probabilities of each class. We have 10 classes to a (10,1) vector is returned. Out of this 10 probabilities the highest one will be the class.

3. **Model Training:**
   There are two datasets with and without augmented images. Model 1 is trained with non-augmented dataset and Model 2 with augmented dataset.

   (a) Forward Propagation
   The input is taken to be batches of images. Each image has 512 features so input matrix X's dimension are (batch size,512). This Z1 is passed to Rectified Linear Unit (ReLU) activation function which returns A1. This A1 is forwarded to next layer. On Z2 softmax activation is applied which return probabilities of each class as A2. The values of A1, A2, Z1 and Z2 are stored to be used for weight updation.
   Equations:
   Z1 = X.W1 + b1.
   Relu activation, A1 = relu(Z1)
   Z2 = A1.W2 + b2
   Softmax activation, A2 = softmax(Z2

   (b) Backward Propagation
   The saved values from the forward propagation A1, A2, Z1 and Z2 are used to update the weights and biases.
   Derivative of softmax, dZ2=A2-Y

   Derivative of W2, dW2 = (A1*dZ2)/l

   Derivative of b2, db2 = $(\sum dZ2)/l$

   Derivative of Z1, dZ1 = relu'(Z1) * dZ2 * W2

Derivative of W1, dW1 = (X1*dZ1)/l

Derivative of b1, db1 = $(\sum dZ1)/l$
Learning rate, lr=0.02

The weights and biases are then updated using learning rate and their derivatives.
W1 = W1 - lr * dW1
W2 = W2 - lr * dW2
b1 = b1 - lr * db1
b2 = b2 - lr * db2

For each batch these weights are updated and passed on to next batch. One weights are updated loss is calculated using binary cross entropy loss function.

(c) Cross entropy loss
Since we aim to classify the images our model will use the cross entropy loss to calculate loss after each backward pass and for each epoch mean loss is calculated.
Loss = (1/l) * $\sum y * \log(ypred)$

4. **Model Evaluation**

(a) **Model 1**
On learning rate as 0.02, batch size as 200 and number of epochs as 50 model 1 is giving an accuracy score of 0.74
Classification report:

	precision	recall	f1-score	support
0	0.81	0.74	0.77	1000
1	0.84	0.85	0.85	1000
2	0.70	0.52	0.59	1000
3	0.53	0.64	0.58	1000
4	0.71	0.68	0.70	1000
5	0.71	0.64	0.67	1000
6	0.74	0.84	0.79	1000
7	0.77	0.79	0.78	1000
8	0.77	0.86	0.82	1000
9	0.84	0.84	0.84	1000
accuracy			0.74	10000
macro avg	0.74	0.74	0.74	10000
weighted avg	0.74	0.74	0.74	10000

Figure 1: Classification report 1

(b) **Model 2**

On learning rate as 0.02, batch size as 200 and number of epochs as 50 model 2 is giving an accuracy score of 0.71

Classification report:

	precision	recall	f1-score	support
0	0.88	0.65	0.75	1000
1	0.65	0.93	0.77	1000
2	0.78	0.48	0.60	1000
3	0.58	0.54	0.56	1000
4	0.72	0.63	0.67	1000
5	0.61	0.72	0.66	1000
6	0.76	0.83	0.79	1000
7	0.66	0.84	0.74	1000
8	0.79	0.80	0.79	1000
9	0.86	0.76	0.81	1000
accuracy			0.72	10000
macro avg	0.73	0.72	0.71	10000
weighted avg	0.73	0.72	0.71	10000

Figure 2: Classification report 2

(c) **Epochs vs loss graph**

We can observe loss of model 1(blue) is comparatively less than the model 2(orange) as the number of epochs increases.

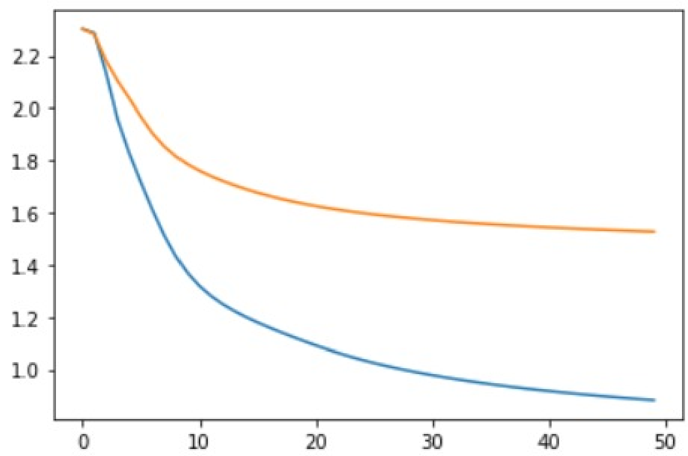

Figure 3: Epochs v/s loss