Machine Learning Homework 8

Hairui Yin

1.

(a) In this part, I define all functions to be called, containing the activation function, initialize parameter function. forward function and backpropagation function. Given only one sample $x = [2, 1], y = 3$, initialize all weights i.i.d uniform(0,1), then train the model 50 epoch with $\gamma = 0.05$. Activation Function sigmoid, and loss is ssquared error.

```python
import numpy as np
import matplotlib.pyplot as plt

def sigmoid(z):
        return 1 / (1 + np.exp(-z))
def sigmoid_derivative(z):
        return (sigmoid(z) * (1 - sigmoid(z)))

def initialize_parameter(input_size, hidden_size, output_size, random=42):
        np.random.seed(random)
        # include bias
        theta1 = np.random.uniform(0, 1, (input_size + 1, hidden_size))
        theta2 = np.random.uniform(0, 1, (hidden_size + 1, output_size))
        return theta1, theta2

def forward(x, theta1, theta2):
        a1 = x
        z2 = np.concatenate((np.array([1]), a1)) @ theta1
        a2 = sigmoid(z2)
        z3 = np.concatenate((np.array([1]), a2)) @ theta2
        return z2, a2, z3

def backward(x, y, z2, a2, z3, theta1, theta2):
        dz3 = z3 - y
        # Gradient for theta2
        dtheta2 = (dz3 * np.concatenate((np.array([1]), a2))).reshape([3,1])
        # Gradient for theta1
        da2 = dz3 * theta2[1:]
        dz2 = da2 * sigmoid_derivative(z2).reshape(2,1)
        dtheta1 = np.outer(dz2, np.concatenate((np.array([1]), x))).reshape([3,2])

        return dtheta1, dtheta2
```

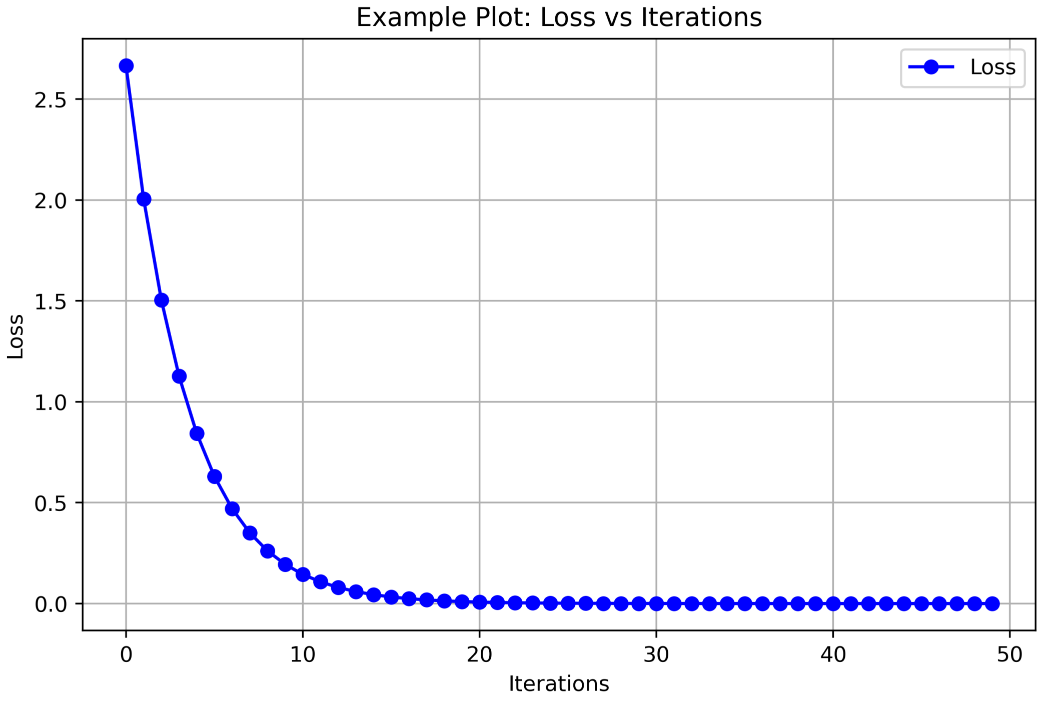
```
theta1, theta2 = initialize_parameter(2, 2, 1)
x, y = np.array([2, 1]), np.array([3])
learning_rate = 0.05

iterations = []
loss = []
for i in range(50):
        z2, a2, z3 = forward(x, theta1, theta2)
        dtheta1, dtheta2 = backward(x, y, z2, a2, z3, theta1, theta2)
        theta1 -= learning_rate * dtheta1
        theta2 -= learning_rate * dtheta2
        # loss record
        iterations.append(i)
        loss.append((y-z3)**2)
```

(b) Since we have recorded squared error loss in question (a), we use the following code to display it:

```
plt.figure(figsize=(8, 5))
plt.plot(iterations, loss, marker='o', linestyle='-', color='b', label="Loss")
plt.title("Loss vs Iterations")
plt.xlabel("Iterations")
plt.ylabel("Loss")
plt.legend()
plt.grid()
plt.show()
```

And the figure is:


Example Plot: Loss vs Iterations
Loss
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
Iterations
Loss