Bài thực hành 2: Hàm Activation, Loss & Accuracy

Câu 1: Viết chương trình tạo 3 ngẫu nhiên nhóm các điểm có dạng xoắn ốc, hình tròn, đường thẳng và đám mây điểm ở những vị trí khác nhau trong không gian 3D. Code mẫu cho tập hợp các điểm có dạng xoắn ốc như bên dưới.

Code mẫu tạo 3 nhóm điểm có dạng xoắn ốc:

```
# Spiral dataset
import numpy as np
import matplotlib.pyplot as plt

#generating a randomized data
N = 100 # number of points per class
D = 2 # dimensionality
K = 3 # number of classes

P = np.zeros((N*K,D)) # data matrix (each row = single example)
L = np.zeros(N*K, dtype='uint8') # class labels
for j in range(K):
  ix = range(N*j,N*(j+1))
  r = np.linspace(0.0,1,N) # radius
  t = np.linspace(j*4,(j+1)*4,N)  + np.random.randn(N)*0.2 # theta
  P[ix] = np.c_[r*np.sin(t), r*np.cos(t)]
  L[ix] = j

# lets visualize the data:
plt.scatter(P[:, 0], P[:, 1], c=L, s=40, cmap=plt.cm.Spectral)
plt.show()
```

Câu 2: Viết một Class Points trong đó cho phép định nghĩa các dataset xoắn ốc, tròn, thẳng và đám mây điểm ở những vị trí khác nhau trong không gian 3D.

Câu 3: Viết 1 class Activation cho phép định nghĩa các hàm Activation sau: Linear Activation, Sigmoid, ReLU, SoftMax.

Câu 4: Trong chương trình Python, viết 1 đoạn code khởi tạo 1 mạng Nơ-ron gồm 2 ngõ vào, 2 lớp Nơ-ron, lớp thứ nhất có 3 nơ-ron và lớp thứ 2 cũng là lớp output có 3 nơ-ron ngõ ra (tượng trưng cho 3 class dữ liệu). Hàm activation của lớp thứ nhất là ReLU, hàm activation của lớp thứ 2 là SoftMax. Khởi tạo dữ liệu xoắn ốc, sử dụng Class Points đã tạo ở câu 2. Đưa dữ liệu của dataset vừa khởi tạo vào đầu vào của mạng NN. Thực hiện feedforward (tính các ngõ ra ở tất cả các lớp). Quan sát các giá trị ở ngõ ra cuối cùng. Kết luận.

Câu 5: Viết chương trình tính hàm loss từ softmax output theo công thức categorical cross-entropy (Là công thức tính loss thông dụng cho các ứng dụng phân loại dữ liệu). Code mẫu như bên dưới:

```
#Loss function examples

import numpy as np
import math
# Output of the NN
softmax_output = [0.7, 0.1, 0.2]
# Ground truth (what we want)
target_output = [1,0,0]
loss = ...
print(loss)
```

Câu 6: Viết 1 class Loss cho phép định nghĩa các hàm Loss và hàm Loss Categorical Cross Entropy (inherit từ hàm Loss). Code mẫu như bên dưới:

```
import numpy as np
import math

# Loss Class
class Loss:
    # Calculate the data and regularization losses given
    # the output and the ground truth values
    def calculate(self,output,y):
        #Calculate sample losses:
```

```
        sample_losses = self.forward(output,y)
        #Calculate mean loss:
        data_loss = np.mean(sample_losses)
        #Return loss:
        return data_loss

# Cross Entropy Loss Class
class Loss_CategoricalCrossentropy(Loss):

    # Forward Pass
    def forward(self, y_pred, y_true):
        samples = len(y_pred)
        # Clip both sides
        y_pred_clipped = np.clip(y_pred,1e-7,1-1e-7) #1e-7 is added to avoid division by 0
        # Probabilities for target values only if categorical labels
        if len(y_true.shape) == 1: # The label array is 1D
            correct_confidence = ...
        elif len(y_true.shape) == 2:  # The label array is 2D
            correct_confidence = ...
            )

        # Losses calculation
        negative_log_likelihoods = -np.log(correct_confidence)
        return negative_log_likelihoods
```

Có thể kiểm tra và quan sát các phép tính của chương trình bằng đoạn code mẫu bên dưới:

```
# Test the class now:
softmax_output = np.array([[0.7, 0.1, 0.2],
                [0.1, 0.5, 0.4],
                [0.02, 0.9, 0.08]])

#class_target = np.array([0, 1, 1]) # cat, dog, dog
class_target = np.array([[1, 0, 0],
                         [0, 1, 0],
```

```
                [0, 1, 0]])

loss_function = Loss_CategoricalCrossentropy()
loss = loss_function.calculate(softmax_output,class_target)
print(loss)
```

Câu 7: Viết đoạn chương trình tính accuracy như sau:

1. Cho ví dụ về dữ kiệu output theo dạng batch
2. Cho ví dụ về vec-tơ Label
3. Từ các Vec-tơ output, tính vec-tơ dự đoán
4. So sánh vec-tơ dự đoán và vec-tơ Label và điền vào mảng giá trị so sánh (chỉ nhận giá trị 0 (T) hoặc 1 (F)), từ đó tính accuracy bằng phép tính trung bình của mảng so sánh này (có thể dùng hàm np.mean).

Câu 8: Trong chương trình Python, viết 1 đoạn code khởi tạo 1 mạng Nơ-ron gồm 2 ngõ vào, 2 lớp Nơ-ron, lớp thứ nhất có 3 nơ-ron và lớp thứ 2 cũng là lớp output có 3 nơ-ron ngõ ra (tượng trưng cho 3 class dữ liệu). Hàm activation của lớp thứ nhất là ReLU, hàm activation của lớp thứ 2 là SoftMax. Sử dụng Loss Categorical Cross Entropy để tính Loss. Khởi tạo dữ liệu xoắn ốc, sử dụng Class Points đã tạo ở câu 2. Đưa dữ liệu của dataset vừa khởi tạo vào đầu vào của mạng NN. Thực hiện feedforward (tính các ngõ ra ở tất cả các lớp). Tính hàm Loss và Accuracy. Quan sát kết quả tính toán. Kết luận.