EN2550 2020: Assignment 04

March 13, 2021

Please note that you must implement the key Python functions and scripts on your own. If you copy from the Internet or others, you will not get the learning experience intended through this assignment.

Dataset: CIFAR10 50,000 $32 \times 32 \times 3$ training images and 10 classes. Accuracy greater than 0.1 shows learning. You can use a deep leaning framework only in item 4, except for loading the dataset as given in the code snippet.

1. A part of the code for a linear classifier for CIFAR10 given in listing 1. For our linear classifier, the score function is $f(x) = Wx + b$, and the loss function is the mean sum of squared errors function. [3 marks]

   ✓ (a) Implement gradient descent and run for 300 epochs.

   ✓ ● (b) Show the weights matrix W as 10 images.

   (c) Report the (initial) learning rate, training and testing loss and accuracies.

   (Hint: If your loss explodes, reduce the leaning rate.)

2. Code a two-layer fully connected network with $H = 200$ hidden nodes. Choose the sigmoid function as the activation function for the hidden nodes. The output layer has no activation function. [3 marks]

   ● ✓ (a) Implement gradient descent and run for 300 epochs.

   (b) Report the (initial) learning rate, training and testing loss and accuracies.

3. ✓ Modify the code in item 2 to carry out stochastic gradient descent with a batch size of 500 ✓ [2 marks]

   (a) Report training and testing loss and accuracies.

   (b) Compare results with item 2 (justify).

4. ✓ Construct a CNN using `Keras.models.Sequential` (with the following configuration: C32, C64, C64, F64, F10. All three convolutions layers are 3×3. Max pooling (2×2) follows each convolution layer. Use SDG (with momentum) with a batch size of 50 ✓ and `CategoricalCrossentropy` as the loss. [2 marks]

   (a) How many learnable parameters are there in this network?

   (b) Report the parameters such as the learning rate and momentum.

   (c) Report training and testing loss and accuracies.

Upload a five-page report named as your_index_a04.pdf. Type out the index number within the file as well. Include important parts of code, results, and interpretations in the file. Your code must be in github, so that we can see the history. We plan to grade this though a viva.

Listing 1: A Part of Linear Classifier for CIFAR10

```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
(x_train, y_train), (x_test, y_test) = keras.datasets.cifar10.load_data()
print('x_train:␣', x_train.shape)

K = len(np.unique(y_train)) # Classes
Ntr = x_train.shape[0]
```

```
Nte = x_test.shape[0]
Din = 3072 # CIFAR10
# Din = 784 # MINIST

# Normalize pixel values
x_train, x_test = x_train / 255.0, x_test / 255.0
mean_image = np.mean(x_train, axis=0)
x_train = x_train - mean_image
x_test = x_test - mean_image

y_train =  tf.keras.utils.to_categorical(y_train, num_classes=K)
y_test = tf.keras.utils.to_categorical(y_test, num_classes=K)

x_train = np.reshape(x_train,(Ntr,Din))
x_test = np.reshape(x_test,(Nte,Din))
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')

std=1e-5
w1 = std*np.random.randn(Din, K)
b1 = np.zeros(K)
print("w1:", w1.shape)
print("b1:", b1.shape)
batch_size = Ntr

iterations =
lr =
lr_decay=
reg =
loss_history = []
train_acc_history = []
val_acc_history = []
seed = 0
rng = np.random.default_rng(seed=seed)
for t in range(iterations):
    indices = np.arange(Ntr)
    rng.shuffle(indices)

    # Forward pass


    # Backward pass

# Printing accuracies and displaying w as images
```