SOI1010 Machine Learning II - Assignment #1

Assigned: Sep. 25, 2023
Due: Oct. 10, 2023 11:59 pm

The submission should include a code (both link to the colab and .py format) and a report that has answers to the questions and results. Use PyTorch (or TensorFlow/JAX). Marks will be deducted if the submission does not include the requested files. **DO NOT** use other libraries, such as scikit-learn/sklearn, to use a model (kNN in this case) you are supposed to implement. **Using sklearn or any other third library or already built-in functions** that you are asked to implement **will result in 0 mark**. Also, if an assignment asks you to implement some model, that means you shouldn't use the built-in implementation from any library for that model in the first place.

Problem #1: Multiclass Classification via k-NN on MNIST [50pts]

MNIST is a database of handwritten digit grayscale images of size 28×28. You cand load MNIST dataset in PyTorch as follows:

```
import numpy as np
import torch
from torchvision import datasets


trainset = datasets.MNIST(root='./data', train=True,
                                        download=True)

testset = datasets.MNIST(root='./data', train=False,
                                       download=True)
```

Data split

You should use 6000 randomly sampled images from the training set for validation. After the data split, you can use DataLoader from PyTorch as follows:

```
# Indices for train/val splits: train_idx, valid_idx
np.random.seed(0)
val_ratio = 0.1
train_size = len(trainset)
indices = list(range(train_size))
split_idx = int(np.floor(val_ratio * train_size))
np.random.shuffle(indices)
train_idx, val_idx = indices[split_idx:], indices[:split_idx]

train_data = trainset.data[train_idx].float()/255.
train_labels = trainset.targets[train_idx]
val_data = trainset.data[val_idx].float()/255.
val_labels = trainset.targets[val_idx]
test_data = testset.data.float()/255.
test_labels = testset.targets
```

(a) Implement an iterative method (using for loop) to classify a single new example. Write down your observations.
(b) Use the broadcasting concept you learned in the laboratory session to classify a single new example. Compare against the result from (a).
(c) Now, implement a k-NN algorithm (starting with k=5) and its training/validation/evaluation code to perform multiclass classification over all digits, using the implementation from (b). Write down your observations.
(d) Improve the algorithm from (c) [Hint: Try to find the desirable distance function, which can be found by googling or going through PyTorch document].
(e) What are the hyperparameters you can tune?
(f) Try at least two other options for each hyperparameter. Report the performance for each option.
(g) You can try more options if you want. What is the final *test* accuracy?